Biblioteczka

Komputer Świat




**NA DVD NARZĘDZIA DO**
- PROGRAMOWANIA
- TWORZENIA MODÓW

**+GOTOWE PROJEKTY**

ISO płyty do pobrania z ksplus.pl

# MINECRAFT

## NAUKA PROGRAMOWANIA PRZEZ ZABAWĘ

Jak za pomocą MakeCode, MCreatora i Pythona zmieniać świat gry i tworzyć mody

## Z TEJ KSIĄŻKI DOWIESZ SIĘ, JAK:

■ łatwo układać programy z gotowych elementów ■ tworzyć i dodawać do gry własne bloki, postaci i broń ■ korzystać ze zmiennych, pętli, list i instrukcji warunkowych ■ posługiwać się językiem programowania Python, by ułatwić sobie rozgrywkę

# KŚ+

## Z TĄ KSIĄŻKĄ E-WYDANIE GRATIS

**Poniżej znajduje się płyta z kodem bonusowym dającym dostęp do e-wydania tej książki w serwisie KŚ+ (ksplus.pl) oraz pliku ISO z cyfrową wersją płyty do pobrania.**

## NA PŁYCIE DVD

**Płyta dołączona do książki zawiera zestaw najlepszych narzędzi programistycznych, w których można tworzyć mody do Minecrafta. Znajdziemy na niej także materiały szkoleniowe – pliki projektów opisanych we wskazówkach, a dodatkowo także polecaną przez redakcję grę Minetest rozwijaną jako wieloplatformowa alternatywa Open Source dla Minecrafta, z modami i serwerami dla wielu graczy.**

**Jeżeli brakuje płyty, poinformuj sprzedawcę lub redakcję: redakcja@komputerswiat.pl**

**Kod bonusowy należy zarejestrować w KŚ+ (ksplus.pl)**

Biblioteczka Komputer Świat

ADRIAN CZECHOWSKI
RADOSŁAW KULESZA
SEBASTIAN LANGA
DAWID LEŚNIAKIEWICZ
PIOTR PEŁKA

MINECRAFT

# NAUKA PROGRAMOWANIA PRZEZ ZABAWĘ

ringier axel springer

**AUTORZY:**

Adrian Czechowski, Radosław Kulesza, Sebastian Langa, Dawid Leśniakiewicz, Piotr Pełka

**REDAKTORZY PROWADZĄCY:** Rafał Kamiński, Agnieszka Al-Jawahiri

**PRZYGOTOWANIE PŁYTY:** Mariusz Michalski

**PROJEKT OKŁADKI:** Robert Dobrzyński

**SKŁAD I ŁAMANIE:** Mariusz Rybak

**KOREKTA:** Jolanta Rososińska

**WYDAWCA:** RINGIER AXEL SPRINGER POLSKA Sp. z o.o.
02-672 Warszawa, ul. Domaniewska 49
tel. 22 7786102
www.ringieraxelspringer.pl

ISBN 978-83-8091-952-5

© Copyright by Ringier Axel Springer Polska Sp. z o.o.
Warszawa 2020

**DYREKTOR WYDAWNICZY:** Paweł Paczuski

**BUSINESS PROJECT MANAGER:** Paweł Bulwan

**DRUK I OPRAWA:** Drukarnia im. Adama Półtawskiego, Kielce

**EGZEMPLARZE ARCHIWALNE:**
literia.pl

prenumerata.axel@qg.com

**E-WYDANIA:** ksplus.pl

**KONTAKT:**
redakcja@komputerswiat.pl

**INTERNET:** komputerswiat.pl, ksplus.pl

Płyta DVD jest dodatkiem do książki

ringier
axel springer

## 1 O CO CHODZI W MINECRAFCIE

Tryby rozgrywki . . . . . 5
Najpopularniejsze moby . . . . . 8
Najpopularniejsze bloki . . . . . 9
Wersje gry Minecraft . . . . . 10

## 2 PROGRAMOWANIE W MAKECODE

Minecraft MakeCode . . . . . 13
Instalacja i przygotowanie platformy – krok po kroku 14
Kategorie bloków podstawowych . . . . . 17
Kategorie bloków zaawansowanych. . . . . 18
Rodzaje bloków . . . . . 19
Kategoria Podstawowe . . . . . 21
Kategoria Gracz. . . . . 23
Kategoria Bloki . . . . . 24
Kategoria Moby . . . . . 25
Kategoria Agent. . . . . 26
Kategoria Rozgrywka . . . . . 27
Kategoria Pozycje. . . . . 28
Kategoria Pętle . . . . . 29
Projekt Arena . . . . . 31
Projekt Parkour . . . . . 34

## 3 MCREATOR: MODY DO MINECRAFTA

Programy do tworzenia minecraftowych modów. . . . . 41
Skąd wziąć MCreatora . . . . . 43
Rozpoczęcie pracy z MCreatorem . . . . . 44
Miniprojekt Blok piór . . . . . 45
Projekt Apokalipsa Creeperów. . . . . 52
Projekt Miecz Niebios . . . . . 62

## 4 JĘZYK PYTHON A MODYFIKOWANIE MINECRAFTA

Python w Minecrafcie: przygotowania. . . . . 71
Teleportacja . . . . . 77
Miniprojekt Skok na bungee. . . . . 82
Tworzenie bloków, brył i konstrukcji. . . . . 83
Miniprojekt Piramida . . . . . 88
Prawda/Fałsz . . . . . 92
Miniprojekt Tor . . . . . 93
Miniprojekt Zapętlona teleportacja . . . . . 95
Miniprojekt Spiralne schody . . . . . 97
Jak znaleźć błąd w skrypcie? . . . . . 99

## DODATKI

Warto wiedzieć . . . . . 102

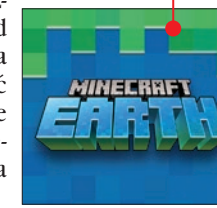
# 1 O co chodzi w Minecrafcie

**Zanim zaczniemy zmieniać świat Minecrafta, ucząc się programistycznych sztuczek, najpierw powinniśmy go poznać. Znajomość rodzajów trybów gry, poznanie mobów i zagadnień związanych z rozgrywką są kluczowe dla zrozumienia działania przedstawionych w dalszych rozdziałach środowisk do modyfikacji tej gry. Oto podstawy rozgrywki oraz najważniejsze postaci**

Najtrudniejszym zadaniem w Minecrafcie jest podobno przetrwanie pierwszej nocy...

Tej gry nie trzeba nikomu przedstawiać. Pomimo upływu lat nadal pod względem popularności plasuje się w ścisłej czołówce w światowych rankingach. Stworzona przez studio Mojang, pojawiła się w internecie 17 maja 2009 roku. Mimo że początkowo wiele jej funkcji było ograniczonych i udostępniała tylko jeden tryb rozgrywki, szturmem zawładnęła sercami fanów na całym świecie. Może o tym świadczyć między innymi fakt, że w ciągu niespełna dwóch miesięcy od oficjalnego wydania gry (18 listopada 2011 roku) Minecraft osiągnął próg miliona sprzedanych kopii wersji na komputery PC. Wróżyło to bardzo dobry start i sprawiło, że słowo „creeper" czy „craftować" na stałe wpisało się do naszego słownika. Chociaż najbardziej rozbudowaną wersją Minecrafta jest właśnie ta na komputery PC, to gra ta ma także wersje na wiele innych urządzeń. Są to przede wszystkim wersje na urządzenia mobilne (iOS, Android, Windows Phone OS / Windows 10 Mobile) oraz na konsole (Xbox 360, Xbox One, PS3, PS4, PS Vita).

Początkowo wersje mobilne (między innymi Bedrock Edition, Pocket Edition) były najmniej rozbudowane w porównaniu z wersjami na inne urządzenia. Wiązało się to głównie z ograniczeniami narzucanymi przez urządzenia mobilne – wiele funkcji było przez nie niedostępnych. Szybki rozwój technologii mobilnych pozwolił jednak na rozwój mobilnego Minecrafta. Dziś możemy na smartfonach i tabletach cieszyć się z porównywalnych funkcji gry co na komputerach i konsolach, a nawet możemy poszerzyć game experience. Przykładem może być Mine craft Earth – gra z rzeczywistością rozszerzoną (augmented reality), pozwalająca tworzyć i budować elementy w świecie rzeczywistym dzięki kamerze urządzenia mobilnego.



# Tryby rozgrywki

Minecraft oferuje kilka trybów rozgrywki. Każdy z nich różni się od innych pod kilkoma względami. Ponadto niektóre tryby nie występują na niektórych urządzeniach. Oto kilka szczegółów na temat trybów rozgrywek.

W Ekwipunku za pomocą opcji wyszukiwania możemy przeglądać i wykorzystywać większość bloków i obiektów

## Tryb: Kreatywny

Tryb Kreatywny został dodany w wersji Beta 1.8 i służy do tworzenia świata. W trybie tym mamy możliwość korzystania niemal ze wszystkich bloków i przedmiotów dostępnych w grze. Pojawiają się one w Ekwipunku.

Tryb Kreatywny pozwala na prawie nieograniczone interakcje ze światem. Bloki, które w innych trybach są niezniszczalne (na przykład skała macierzysta), w tym trybie można łatwo zniszczyć. Gracz ma umiejętność latania, którą można aktywować przez podwójne wciśnięcie klawisza [spacja]. Niektóre elementy są dostępne wyłącznie w trybie Kreatywnym i nie znajdziemy ich w innych trybach (na przykład jaja przyzywające moby). Gracz jest w zasadzie nieśmiertelny, a wszelkie moby, nawet agresywne, nie są w stanie wejść z graczem w interakcję negatywną.

### Z RAPORTU STEVE'A*

**W trybie Kreatywnym nie znajdziemy pasków pancerza czy punktów życia. Powodem jest nieśmiertelność gracza. Istnieją jednak sposoby na śmierć gracza w trybie Kreatywnym:**

- **komenda /kill**
- **zejście poniżej poziomu -64 wprożnię (Java Edition)**

## Tryb: Przetrwanie

Tryb Przetrwanie to obok trybu Kreatywnego najbardziej popularny typ rozgrywki. Gracz zaczyna tu swoją przygodę od zera – nie ma przedmiotów, a jego podstawowym celem jest przetrwanie. Z tego powodu należy zbierać surowce, unikać agresywnych mobów lub walczyć z nimi, wytwarzać narzędzia, bloki, a także zdobywać i przetwarzać pożywienie. Jest oczywiście

* Steve to główny bohater gry. W raportach Steve'a w tej książce znajdziemy ciekawostki i przydatne informacje dotyczące świata Minecrafta.

o wiele więcej elementów rozgrywki. Gracz może wybrać jeden poziom trudności gry spośród czterech. Są to:

- Pokojowy
- Łatwy
- Trudny
- Hardcore

Poziom Hardcore jest w zasadzie podobny do poziomu Trudnego. Ważne jest jednak to, że występuje on tylko w Java Edition i po śmierci gracz nie odradza się, ale zamiast tego przechodzi w tak zwany Spectator mode (czyli tryb Widza), umożliwiający obserwację dalszej rozgrywki, ale bez możliwości wchodzenia w jakąkolwiek interakcję z otaczającym światem. Dlatego też jest on bardzo często wykorzystywany w grach multiplayer (gdy gra wielu graczy).

Widok ekwipunku w trybie Przetrwania

## Z RAPORTU STEVE'A

**Ph1lza** – rekordzista poziomu Hardcore. Gracz o tym nicku utrzymywał się przy życiu przez pięć lat regularnej rozgrywki, którą przedstawiał na swoich streamach.

## PASKI INFORMUJĄCE O NAJWAŻNIEJSZYCH DANYCH DOTYCZĄCYCH ROZGRYWKI

**A pasek pancerza** – graficznie przedstawiony w formie płytek napierśnika, informuje o poziomie pancerza. Najwyższy poziom osiągnąć można przy użyciu diamentowej zbroi

**B pasek życia** – informuje o pozostałych punktach życia

**C pasek powietrza** – w formie półprzezroczystych niebieskich pęcherzyków mówi, ile powietrza zostało nam w płucach – występuje tylko w przypadku zanurzania się

**D pasek głodu** – gdy spadnie do zera, powoduje utratę punktów życia. Aby go uzupełnić, należy zjeść pożywienie (każdy rodzaj pożywienia przywraca inną liczbę punktów)

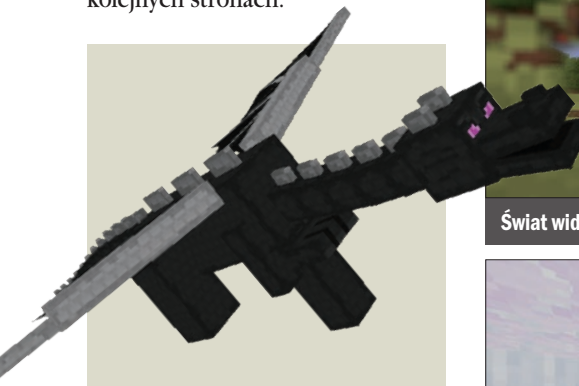
**E pasek doświadczenia** – pozwala określić, ile punktów doświadczenia brakuje nam do kolejnego poziomu

## Tryb: Przygoda

Kolejnym, trochę mniej rozbudowanym trybem jest tryb Przygody. Charakterystyczną cechą tego trybu jest to, że blok nie może być zniszczony, jeśli nie mamy odpowiednich narzędzi do tego celu.
Tryb ten jest również dostosowany do gry wieloosobowej.

## Tryb: Widz

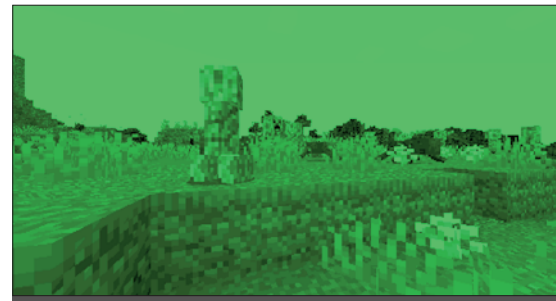
Tryb ten umożliwia bierną obserwację świata gry. Pozwala on na wiele ciekawych opcji niedostępnych w innych trybach, na przykład przelatywania przez solidne bloki. Gracz ma również nieśmiertelność, jednak tak jak w trybie Kreatywnym próżnia jest w stanie spowodować śmierć gracza.
Najciekawszym elementem jest jednak możliwość oglądania świata z pozycji mobów Minecrafta, na przykład Creepera **A**, Pająka Jaskiniowego **B** czy Endermana **C**.
Więcej o mobach przeczytamy na kolejnych stronach.



## Z RAPORTU STEVE'A

**Jedynym mobem, z którego perspektywy nie możemy zobaczyć świata, jest Smok Endu (Ender Dragon), pierwszy oficjalny boss, jaki pojawił się w Minecrafcie.**

Standardowy tryb widzenia



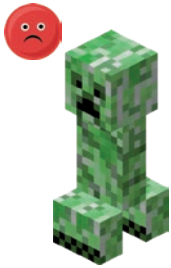
Świat widziany oczami Creepera **A**

Świat widziany oczami Pająka Jaskiniowego **B**

Świat widziany oczami Endermana **C**

# Najpopularniejsze moby

Moby to postaci sterowane przez komputer, które stanowią nieodłączny element Minecrafta. Są to wszystkie poruszające się, „żyjące" postaci w świecie Minecrafta. Moby dzielimy przede wszystkim na agresywne (zawsze atakują gracza), neutralne (atakują tylko sprowokowane) i pasywne (nigdy nie atakują). Każdy z nich ma swoją sztuczną inteligencję oraz wchodzi z graczem w różne interakcje.

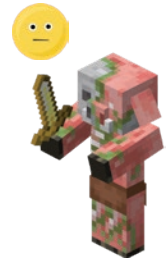


**Creeper** – symbol Minecraf ta. Jest to agresywny mob, który wybucha, gdy znajdzie się zbyt blisko gracza, wydając przy tym charakterystyczny dźwięk. Po uderzeniu przez piorun przyjmuje formę naładowanego Creepera, którego siła wybuchu jest dwukrotnie większa (siła wybuchu zwiększa się z 3 do 6).

**Pająk** to także bardzo popularny mob w Minecrafcie. Atakuje gracza w nocy i w zaciemnionych miejscach, na przykład w Puszczy. W innych przypadkach jest neutralny i atakuje tylko wtedy, gdy zostanie sprowokowany (na przykład zaatakowany przez gracza). Jest to najszybszy mob występujący w grze.

**Zombie** to najczęściej spotykany mob agresywny, występujący w grze. W zależności od warunków i biomu występowania będzie pojawiał się w różnych wariantach – Posucha (Pustynia), Utopca (pod wodą) czy też Zarażonego Osadnika (zainfekowany Osadnik, który miał kontakt z Zombie). Po zabiciu pozostawia po sobie zgniłe mięso.

**Enderman** to neutralny mob, który pojawia się na wszystkich terenach niezależnie od Wymiaru. Atakuje gracza, gdy ten go sprowokuje – poprzez zaatakowanie lub spojrzenie mu w oczy z odległości mniejszej niż 64 bloki. Enderman ma możliwość teleportacji i jest odporny na bronie długodystansowe, takie jak łuk.



**Zombie Pigman** to mob naturalnie występujący jedynie w wymiarze Netheru. Atakuje tylko wtedy, gdy sam zostanie zaatakowany. Wśród tych mobów funkcjonuje zachowanie stadne. Zaatakowanie jednego Zombie Pigmana sprawia, że w odwecie atakuje gracza całe stado. Zombie Pigman może się pojawić również w świecie podstawowym, gdy w świnię uderzy piorun.

**Żelazny Golem** to mob występujący w naturalnie wygenerowanych wioskach. Jego zadaniem jest obrona mieszkańców wios ki, dlate go atakuje gracza, gdy ten sprowokuje go atakiem na Osadnika. Żelazny Golem może być stworzony przez gracza. Nie pojawia się w wiosce Zombie.

Oznaczenia mobów: – pasywny, – neutralny, – agresywny.

**Kurczaka** spotkać można najczęściej błąkającego się bez celu po okolicy. Jest to mob, którego gracz może bardzo łatwo aktywnie hodować. Dostarcza piór, które są niezbędne do stworzenia strzał, jaj oraz mięsa. Może występować w grupach, chociaż najczęściej spotyka się pojedyncze osobniki.

**Kałamarnica** jest stworzeniem wodnym, które w zasadzie nie reaguje na obecność gracza. Ucieka, gdy zostanie zaatakowana. Nie przepada za Strażnikami i jest przez nich atakowana. Pozostawia po sobie atrament służący między innymi do zabarwiania przedmiotów i obiektów w kategorii Barwnik.

**Ocelota** nie znajdziemy w żadnym innym biomie poza Dżunglą. Jest to bardzo ostrożny mob, który ucieka przed graczem, a atakuje tylko małe żółwie i kurczaki. Oceloty są dobrymi pomocnikami w walce z Creeperami, które przed nimi uciekają.

# Najpopularniejsze bloki

**Ziemia** to najpopularniejszy po powietrzu i kamieniu blok w Minecrafcie.

**Kamień** jest podstawowym blokiem. Wydobycie go prowadzi do pozyskania bruku. Stosowany do produkcji prostych narzędzi.

**Piasek** to blok występujący w pobliżu zbiorników wodnych. Stosowany głównie do produkcji szklanych wyrobów.

**Drewno** to blok, którego wygląd uzależniony jest od biomu, w którym występuje. To jeden z podstawowych bloków w craftingu.

**Żwir** jako pospolity blok może być znaleziony zarówno w świecie podstawowym, jak i w Netherze. Działa na niego grawitacja, co wywołuje spadanie, gdy usunie się wszelkie bloki pod nim.

**Ruda węgla** jest powszechnie występującym blokiem, który pozwala na wydobywanie węgla. Węgiel może być wydobyty dowolnym kilofem.

**Ruda złota** służy do pozyskiwania złota. Jest to najrzadsza z rud – szansa jej wystąpienia to tylko 0,1437 procent.

**Ruda żelaza** po przetopieniu daje żelazo. Jedna z bardziej pospolitych rud. Występuje na wysokości od 1 do 63.

**Ruda diamentu** to jedna z najcenniejszych i najrzadszych rud. Jej największa koncentracja występuje na poziomach 11 i 12. Jest przetapiana na diament.

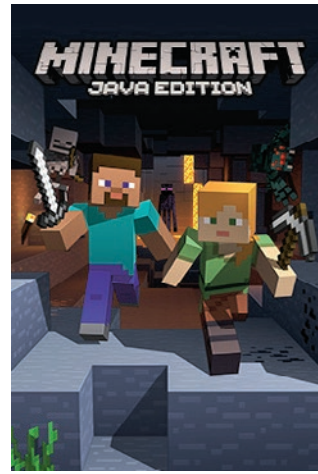
**A oto list gończy za Minecraftem ułożony przez autorów książki – wszystkie najważniejsze cechy tej popularnej gry na jednym plakacie**

# Wersje gry Minecraft

Minecraft zdobył wyjątkowo dużą popularność i rzesze fanów na całym świecie.

Dlatego producenci gry musieli odpowiedzieć na zapotrzebowanie na dostępność Minecrafta na różnych platformach oraz urządzeniach, w tym także mobilnych, bo są one powszechnie używane.

I dzięki temu dzisiaj podczas naszych minecraftowych przygód możemy się spotkać z następującymi wersjami tej gry.




## Minecraft Java Edition

Jest to pełna wersja gry oraz jej oficjalna wersja na komputery PC. Obecna wersja gry to 1.15. Dostępne są również różne podwersje. Użytkownicy mogą brać udział w testowaniu nowych wersji gier poprzez korzystanie z tak zwanych snapshotów – wersji testowych udostępnianych cyklicznie. Jest dostępna zarówno na system operacyjnym macOS, Linux, jak i Windows.

## Minecraft for Windows 10

Wersja przeznaczona wyłącznie do systemu operacyjnego Windows 10. Umożliwia wieloplatformową rozgrywkę na różnych urządzeniach.

## Minecraft Education Edition

Wersja zaprojektowana specjalnie do wykorzystania w szkołach podczas nauki programowania. Ciekawymi funkcjami tej edycji są między innymi identyfikatory uczniów, samouczki, mapy ze współrzędnymi itp. Opiera się na wersji Minecraft for Windows 10.

## Minecraft Bedrock Edition

Wersja na smartfony i tablety z systemem Android, iOS, Windows Phone i Windows 10. Ta edycja łączy różne poprzednie wersje na smartfony (Pocket Edition, Gear VR Edition itp.) oraz konsole (Xbox One, Xbox 360, PS3, PS4, PSVita, Wii, Nintendo Switch). Bedrock Edition ma pewne ograniczenia, na przykład brakuje w niej różnych bloków oraz niektórych elementów Survivalu (tryb Przetrwanie).

## DEKALOG PRZETRWANIA

1. Pierwsza noc jest najtrudniejsza – upewnij się, że dasz radę na czas znaleźć czy zbudować sobie schronienie.

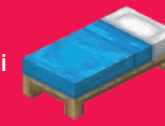
2. Stwórz Crafting Table i pierwsze przedmioty najszybciej, jak się da.

3. Unikaj wrogich mobów i staraj się z nimi nie walczyć.

4. Postaraj się zorganizować sobie oświetlenie.

5. Znajdź źródło pożywienia (jabłko, kurczak itp.).

6. W miarę możliwości stwórz łóżko.



7. Dokładnie nasłuchuj i obserwuj okolicę, szczególnie nocą.

8. Nie biegaj na oślep – możesz wylądować na przykład w głębokim wąwozie.

9. Zwracaj uwagę na słońce, aby wiedzieć, czy zbliża się zmrok.

10. Regularnie sypiaj – brak snu może przywołać Fantoma.

# 2 Programowanie w MakeCode

**Znamy już świat Minecrafta, teraz zacznijmy go zmieniać, na początek – korzystając z platformy MakeCode. Za jej pomocą wykonamy w grze własny tor przeszkód oraz arenę**

Świat Minecrafta daje wiele możliwości różnorodnej rozgrywki. W zależności od trybu czy poziomu trudności możemy w dowolny sposób pokierować swoim losem. Szczególnie dużo możliwości daje tryb Kreatywny. Wraz z rozwojem gry pojawiło się jednak także bardzo dużo nieoficjalnych dodatków i modów (modyfikacji), których zadaniem jest urozmaicenie rozgrywki.

Są to zarówno mody techniczne (wprowadzające nowe mechaniki i procesy), jak i mody przygodowe (dodające nowe potwory, struktury) czy też mody graficzne (wpływające na wizualne przedstawienie świata).

Chęć stosowania modyfikacji wynika głównie z wielkiej potrzeby graczy, by zwiększać i poprawiać gaming experience (wrażenia z rozgrywki). Gracze chcą mieć wpływ na grę oraz możliwość zmiany jej mechaniki, dlatego coraz częściej sięgają nie tylko po różnego rodzaju mody, ale także po różne aplikacje i programy zewnętrzne, które pozwalają najpierw na samodzielne stworzenie jakiegoś rozwiązania, a później na zaimplementowanie go w rozgrywce.

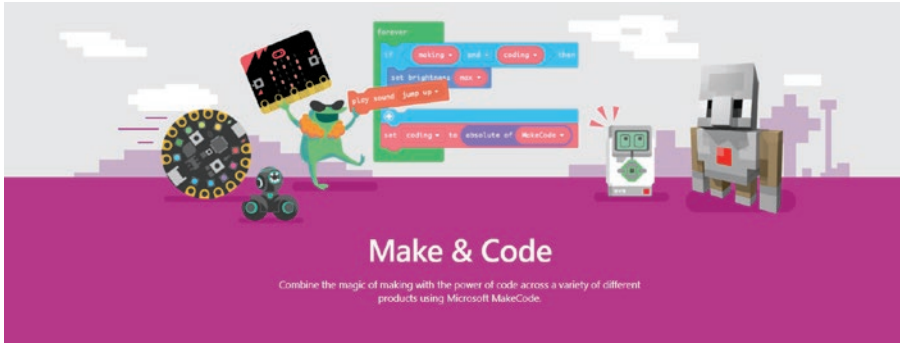
W trakcie lektury tej książki poznamy trzy najpopularniejsze sposoby zewnętrznej ingerencji w świat Minecrafta. Pierwszy z nich to platforma **MakeCode**, która za pomocą blokowego języka programowania pozwala stworzyć wiele komend. Następnie, łącząc się z Minecraftem przez specjalny konektor, można wprowadzić zmiany do świata Minecrafta. Kolejnym sposobem będzie poznanie podstaw tworzenia modów za pomocą **MCreatora** – programu pozwalającego na łatwe

i szybkie tworzenie prostych modów wykorzystywanych później w grze.
Ostatni sposób, jaki poznamy w książce, to będzie wykorzystanie języka programowania **Python** do ingerencji w świat Minecrafta.
Wszystkie te sposoby zostaną przedstawione krok po kroku, a specjalnie przygotowane praktyczne lekcje pozwolą na utrwalenie pozyskanej wiedzy. Możemy zmieniać świat Minecrafta po swojemu. Zaczynajmy!

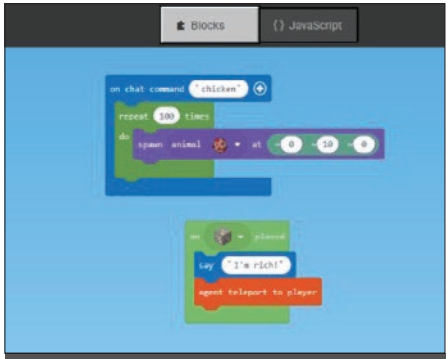
### POPULARNE MODY DO MINECRAFTA WEDŁUG PORTALI RECENZENCKICH

1. Galacticraft
2. Pixelmon
3. Infamy
4. The Lost Cities
5. Hunting Dimension
6. Battle Towers

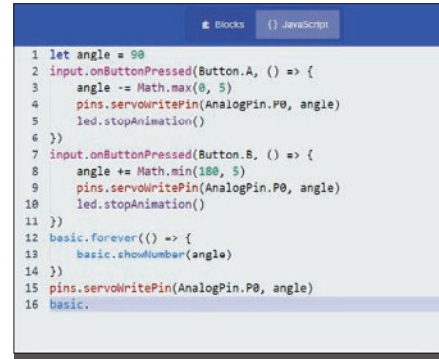
# Minecraft MakeCode




**Minecraft MakeCode** jest to darmowa platforma pozwalająca na tworzenie różnego rodzaju skryptów, które można zaimplementować do rozgrywki. Udostępnia możliwość programowania zarówno w języku blokowym, jak i w języku JavaScript.
Po skonstruowaniu odpowiednich skryptów należy przenieść je do Minecrafta. Stosowany



Edytor blokowy w Minecraft MakeCode



Edytor JavaScript w Minecraft MakeCode

jest do tego specjalny przenośnik – **Code Connection for Minecraft**. Dzięki niemu jesteśmy w stanie połączyć Minecrafta ze środowiskiem, platformą, w której tworzymy skrypty (Code Connection for Minecraft obsługuje również inne platformy, na przykład środowisko **Tynker**).

Code Connection for Minecraft jest dostępny między innymi w Microsoft Store

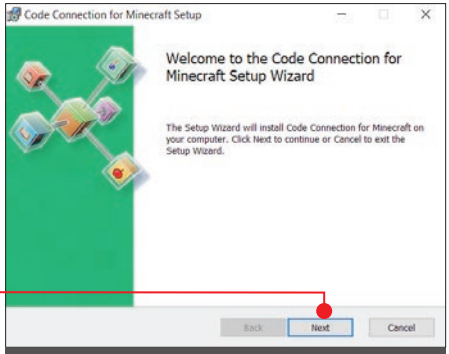
# Instalacja i przygotowanie platformy – krok po kroku

**1 Instalacja Minecraft for Windows 10 lub Minecraft Education Edition**

Instalujemy **Minecraft for Windows 10** lub **Minecraft Education Edition** (DVD-KOD: 005, na płycie dołączonej do książki jest wersja próbna). Możemy także pobrać **Minecraft Education Edition** ze strony **education.minecraft.net/download** Postępujemy zgodnie z procedurą instalacyjną.

**WAŻNE!**

Minecraft MakeCode nie jest obsługiwane przez Java Edition.



Ekran startowy instalacji dla Code Connection for Minecraft

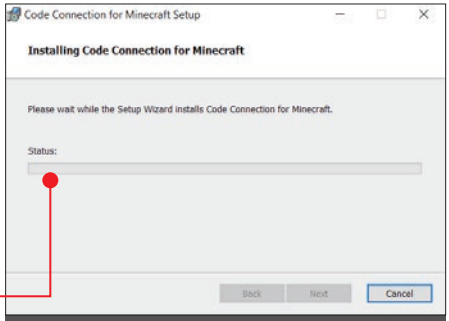
**2 Instalacja Code Connection for Minecraft**

**Code Connection for Minecraft** pobieramy ze Sklepu Microsoft (program jest też na płycie – DVD-KOD: 001).

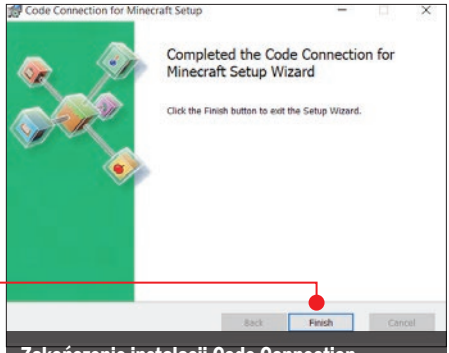
By rozpocząć instalację, należy dwukrotnie kliknąć na plik **CodeConnection.msi**.

Potem klikamy na przycisk **Next**. Program zostanie automatycznie zainstalowany. O postępie instalacji będzie nas informować pasek postępu.

Po zakończeniu instalacji zobaczymy ekran końcowy, na którym wystarczy kliknąć na przycisk **Finish**.



Drugi etap procesu instalacyjnego dla Code Connection for Minecraft



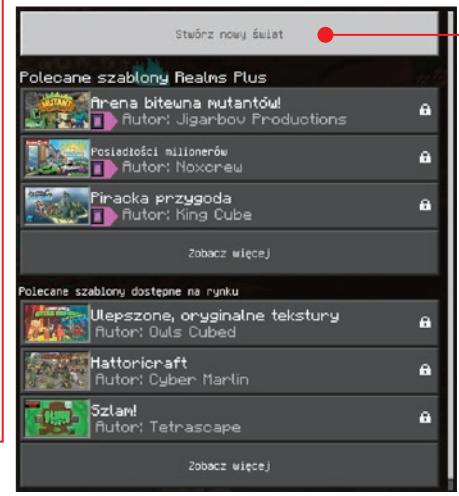

**Zakończenie instalacji Code Connection for Minecraft**

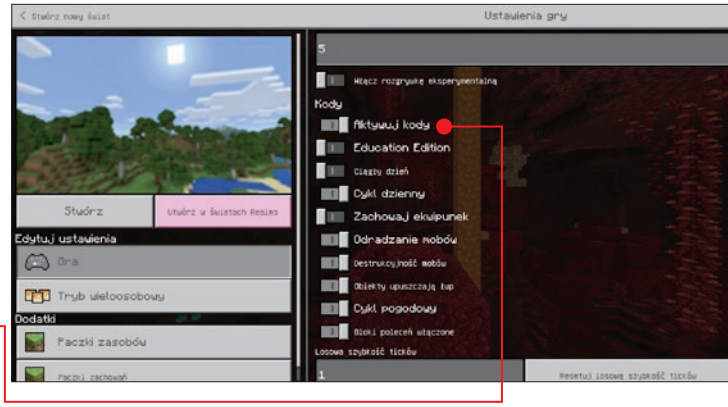
## 3 Stworzenie nowego świata

Aby rozpocząć pracę z Minecraft MakeCode, należy najpierw uruchomić grę i stworzyć nowy, prosty świat. Na własnym, stworzonym przez siebie świecie będziemy testować, a potem dodawać skrypty.

Zaczynamy od wybrania w głównym menu opcji **Graj**. W nowym oknie wybieramy opcję **Stwórz nowy**, a potem **Stwórz nowy świat**, co pozwoli przejść do wyboru opcji związanych z tworzeniem świata.

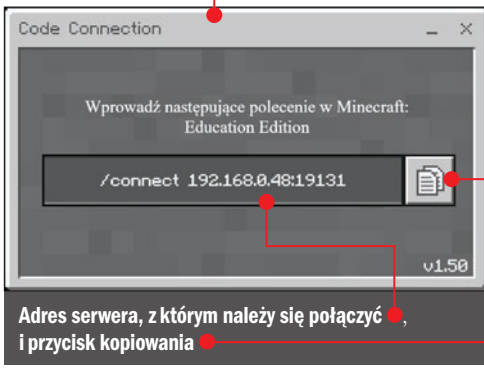
Na kolejnym etapie wprowadzamy ustawienia dla świata – takie jakie nam odpowiadają, tylko jedno ustawienie jest obowiązkowe: musimy włączyć kody za pomocą opcji **Aktywuj kody**. Umożliwi to wpisywanie komend i łączenie się z platformą za pomocą Code Connection for Minecraft. Na koniec klikamy na **Stwórz**.



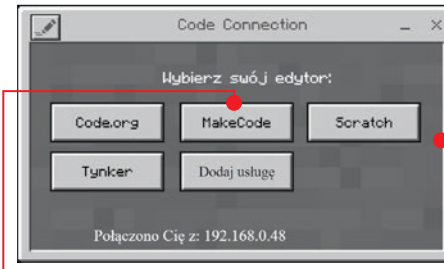



## 4 Uruchomienie Code Connection

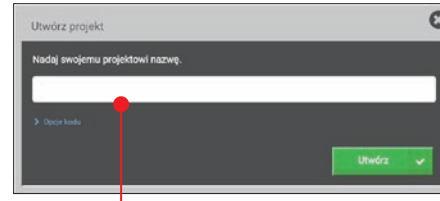
Po włączeniu i skonfigurowaniu Minecrafta przyszedł czas na konfigurację Code Connection. Po uruchomieniu programu zobaczymy okno. Kopiujemy z niego adres serwera. Kolejnym krokiem jest wejście do uruchomionej gry i naciśnięcie klawisza T, aby przejść do konsoli tekstowej. Wklejamy adres i naciskamy enter.

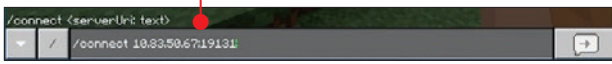



Adres serwera, z którym należy się połączyć, i przycisk kopiowania

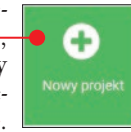



Code Connection zapyta nas wtedy, z którą platformą chcemy się połączyć. Po wybraniu **MakeCode** proces konfiguracji zostanie zakończony. Teraz możemy już do woli modyfikować świat Minecrafta poprzez własne skrypty.




Klikając na zielony przycisk, tworzymy nowy projekt i nadajemy mu nazwę. Dzięki temu zostaniemy przeniesieni do okna, w którym odbywa się tworzenie skryptów i programów.



Cześć, jestem Agentem. Pomogę ci budować świat i będę twoim stróżem. Możesz dowolnie mnie programować. Cieszę się, że będziemy razem pracować.

Poznajmy Agenta – postać, która będzie nam towarzyszyć (to mob występujący tylko w Education Edition)

# Kategorie bloków podstawowych

W tej kategorii znajdziemy najczęściej używane bloki. Jest to szybki dostęp do najważniejszych bloków.

Modyfikowanie ustawień całego świata (cykl dnia i nocy, pogoda, czas) oraz zmiana trybów gry.

Znajdują się tu wszelkie bloki odpowiedzialne za zachowanie gracza w grze – teleportacje, uruchamianie określonego elementu, gdy wykonamy ruch, itp.

Kategoria programistyczna zarządzająca powtarzalnością czynności (dzięki zawartym w niej blokom możemy sprawić, by dana czynność się powtarzała).

Kategoria odpowiedzialna za architekturę krajobrazu. Dzięki niej jesteśmy w stanie zmieniać już istniejący świat Minecrafta, na przykład edytować biomy.

Kategoria pozwalająca na edytowanie zmiennych. Zmienną można porównać do pudełka, w które wkładamy jakiś element i ten element może się zmieniać. O zmiennych dowiemy się więcej z dalszej części książki.

Dodawanie mobów, edycja ich zachowań oraz wszelkie elementy związane z postaciami sterowanymi przez komputer.

Działania na lokalizacji w grze czy na koordynatach X, Y i Z.

Edycja zachowania Agenta i programowanie jego ruchów do budowy obiektów, tworzenia przedmiotów i modyfikacji otaczającego świata.

Kategoria programistyczna, dzięki której możemy tworzyć między innymi konstrukcje warunkowe, na przykład: jeżeli podejdziemy do jajka, to spadnie deszcz kurczaków.

Kategoria programistyczna odpowiedzialna za działania matematyczne, jak dodawanie.

# Kategorie bloków zaawansowanych

Poza blokami podstawowymi są dostępne również bloki zaawansowane, których znajomość przydaje się przy tworzeniu bardziej skomplikowanych elementów i programów. Dotyczą one kwestii czysto programistycznych. Bloki te z pewnością przydadzą się nam na dalszych etapach nauki programowania. Oto ogólny opis tych bloków.

Spośród bloków zaawansowanych z pewnością wykorzystamy Kształty. Przygotuj się na odrobinę geometrii!

Funkcje nazywane są często podprogramami. Są to fragmenty kodu, który może być wykonywany wielokrotnie przez różne elementy programu.

Funkcja podobna do Agenta. Pozwala jednak na dużo bardziej zaawansowane i skomplikowane sterowanie budowaniem struktur. Dzięki konstruktorowi można dużo szybciej stworzyć takie struktury, jak zamki czy chatki.

Ta kategoria pozwala na pracę z tablicami. Tablice są sposobem organizacji wielu zmiennych najczęściej tego samego typu. Korzystając z tablicy, można na przykład stworzyć listę mobów, które będą atakowały nas podczas gry.

Kształty umożliwiają dodanie gotowych kształtów, takich jak kula czy rząd. Ułatwiają tworzenie elementów geometrycznych.

Kategoria, która umożliwia wprowadzanie tekstu i operowanie nim w programie. Na przykład możemy dzięki niej nazywać funkcje czy zmienne.

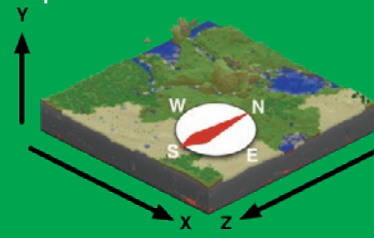
Rozszerzenia pozwalają wykorzystać specjalną platformę do programowania, na przykład robotów czy urządzeń elektronicznych, a także używać jej we współpracy z innymi platformami.

## LOKALIZACJA W MINECRAFCIE

Istotną rolę w świecie Minecrafta odgrywa lokalizacja gracza na mapie. Za lokalizację postaci na mapie odpowiadają koordynaty X, Y, Z. Dzięki nim możemy określić na przykład miejsce respawnu (po przespaniu się w łóżku staje się ono miejscem spawnu po przegranej). Są też użyteczne, gdy chcemy zapamiętać, gdzie znajdowała się wioska, piramida czy inny obiekt.

X i Z pozwalają na określenie, gdzie znajduje się postać na mapie płaskiej, natomiast Y to wysokość.

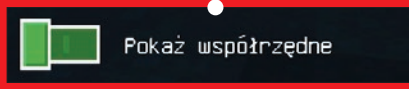
W grze szczególną rolę odgrywa wysokość, która warunkuje generowanie różnych rud surowców na danym poziomie wysokości, na przykład diament nigdy nie występuje powyżej wysokości 20 warstw (warstwa to wysokość jednego bloku w grze liczona od minimalnej wysokości, na której występuje skała macierzysta). Dzięki temu nigdy nie znajdziemy diamentu w powietrzu.




## UWAGA!

Aby sprawdzić koordynaty, należy nacisnąć podczas rozgrywki klawisz F3.

W przypadku niektórych wersji gry (na przykład Minecraft for Windows 10), aby uruchomić współrzędne, należy wykorzystać do tego opcję w **Ustawieniach świata**.




## Z RAPORTU STEVE'A

Świat Minecrafta ma swoje granice. Maksymalna wysokość to 256 bloków, natomiast granica w świecie płaskim to 30 milionów bloków, licząc od środka mapy. Wyjście poza granice mapy sprawia, że gracz dusi się, choć w pierwotnych wersjach gry istniały tam odległe lądy, czyli miejsca, gdzie gra „psuła się" i można było na przykład widzieć moby przenikające przez bloki.

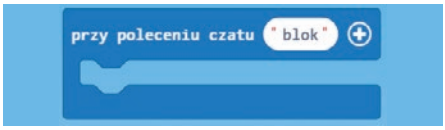
# Rodzaje bloków

Podczas nauki programowania poznamy różne rodzaje bloków. Niektóre są kluczowe dla działania programów, a inne są tylko pewnym uzupełnieniem już istniejących skryptów. W dalszej części książki znajdziemy przykłady zastosowania wielu bloków i sprawdzimy, jak działają one w świecie Mine crafta. Stworzymy też samodzielnie, krok po kroku, dwa projekty. Pomocą i wskazówkami może nam służyć Agent.

Rodzaje bloków w MakeCode różnią się pewnymi elementami i nie mogą być używane w ten sam sposób.

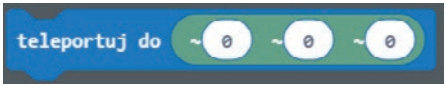
Podstawową różnicą, jaką na pewno zauważymy, jest charakterystyczny kształt każdego z rodzajów bloków.

## Bloki ogólne




To rodzaj bloków, które stanowią najczęściej przestrzeń dla innych bloków. Do jego wnętrza wprowadza się inne bloki. Bez bloków ogólnych inne bloki nie są w stanie funkcjonować. Są to jednocześnie bloki inicjujące (startowe) i mają bardzo duży wpływ na tak zwany przepływ programu. Bloki ogólne można porównać do wielkiego domu, w którym umieszczamy przeróżne meble, takie jak stół czy krzesła.

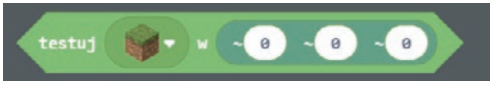
## Bloki strukturalne



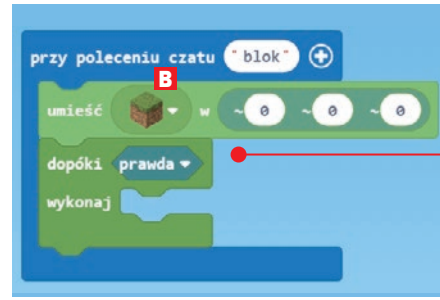

Bloki te stanowią wypełnienie bloków ogólnych i bez nich nie są w stanie wykonywać swojej funkcji. Mogą łączyć się z innymi blokami w wielkie struktury. To najczęściej występujący rodzaj bloku w MakeCode. Bardzo często zawierają różnego rodzaju przestrzenie, w których można umieszczać inne bloki.

## Bloki uzupełniające

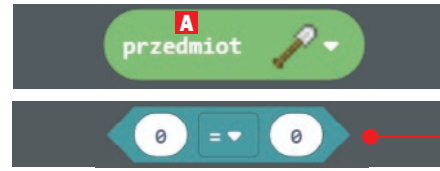
Bloki te nie są w stanie funkcjonować samodzielnie. Ich




działanie opiera się na uzupełnianiu innych bloków o dodatkowe, szczegółowe informacje. Najczęściej mają zaokrąglone lub ostre boki. Z tego powodu bloki te mogą uzupełniać tylko konkretne miejsca w blokach strukturalnych.
Na przykład blokiem **przedmiot** A można zastąpić zaokrąglone miejsce B, zmieniając charakter bloku strukturalnego. W przykładzie na ilustracji program nie umieszczałby

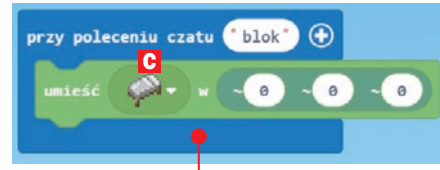



już bloku porośniętej ziemi w miejscu **0, 0, 0** (czyli tam, gdzie stoi gracz), tylko umieściłby

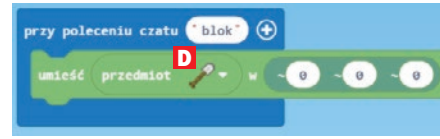



tam przedmiot łopatę. Podobnie jest w przypadku bloków o zaostrzonych końcach ●.
Można więc zauważyć, że dodanie czy zmiana bloku uzupełniającego w bloku strukturalnym może całkowicie zmienić jego znaczenie.

**Przykład:**




Powyższy skrypt ● sprawia, że łóżko C zostanie umieszczone w koordynatach **0, 0, 0**, czyli w miejscu, w którym stoi gracz. Dodajemy do niego blok uzupełniający.
W rezultacie otrzymamy poniższy skrypt ●, który zamiast łóżka będzie umieszczał w koordynatach **0, 0, 0** łopatę D.
Wiele bloków ma również dodatkowe opcje, takie jak listy rozwijalne czy białe pola z liczbami. Poznamy je na kolejnych stronach książki.

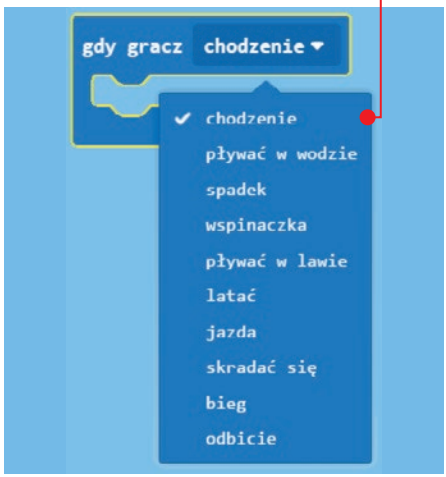


# Kategoria Podstawowe

PODSTAWOWE

Dzięki blokom podstawowym można się bardzo wiele dowiedzieć na tematy związane z podstawami programowania. Dzięki nim poznamy również dodatkowe opcje ukryte w wielu blokach, których znajomość jest kluczowa w tworzeniu własnych skryptów.

Dobrym przykładem są listy rozwijane. Znajdują się one we wszystkich blokach mających białą strzałkę. Po kliknięciu na strzałkę pojawia się lista dostępnych opcji.

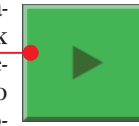


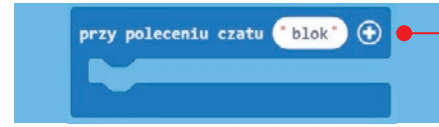

Powyższy blok sprawi, że wszystkie inne bloki umieszczone w jego wnętrzu będą wykonywać się tylko wtedy, gdy gracz wykona wskazaną spośród czynności z listy.

**Przykład:**

W pierwszym przypadku A ogień będzie rozmieszczany w miejscu przebywania bohatera podczas chodzenia, natomiast w drugim przypadku B ogień będzie pojawiać się dopiero po rozpoczęciu biegu.

Aby uruchomić kod z przykładu, należy kliknąć na przycisk ze strzałką w prawo na zielonym tle. Wtedy, wracając do włączonego Minecrafta, możemy sprawdzić działanie naszego skryptu. Po uruchomieniu gry podczas chodzenia lub biegania będą pojawiać się płomienie.






Innym przykładem bloku podstawowego jest blok komendy tekstowej.

Bloki zawarte w środku tego bloku wykonają się dopiero, gdy otworzymy czat. Wpisując komendę, uruchomimy skrypt.

## UWAGA!

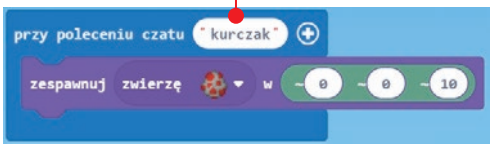
**Aby uruchomić czat, naciskamy klawisz T i wpisujemy komendę. Klawisz / wywołuje konsolę potrzebną do połączenia się z MakeCode, nie używamy jej do uruchamiania skryptów.**

**Przykład:**

Oto skrypt odpowiedzialny za umieszczenie bloku lawy na wysokości 10 bloków. Wykona się on jednak dopiero wtedy, gdy zostanie wpisana w konsoli czatu komenda **blok**.

W efekcie otrzymamy blok lawy pojawiający się nad naszym bohaterem na wysokości 10 bloków.

W analogiczny sposób blok komendy tekstowej można wykorzystać, by pojawiał się na przykład kurczak. Oto efekt w grze.



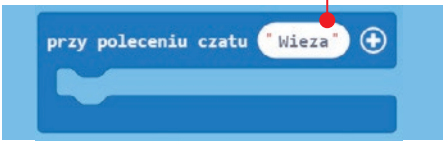
## Miniprojekt Wieża

Spróbujmy teraz wykorzystać zdobytą wiedzę i stworzyć niewielką wieżę z bloków skały macierzystej.

## Z RAPORTU STEVE'A

Lokalizacja **0, 0, 0** to lokalizacja gracza w grze. Jeżeli spróbujemy postawić jakiś blok w lokalizacji **0, 0, 10**, to pojawi się on w odległości 10 bloków od naszej postaci. Jeśli przesuniemy się później na przykład o 100 bloków w inną stronę i znowu postawimy blok w lokalizacji **0, 0, 10**, to nowy blok pojawi się znowu w odległości 10 bloków od naszej postaci.

**1** Dodajemy blok komendy tekstowej i edytujemy komendę na **Wieza**.

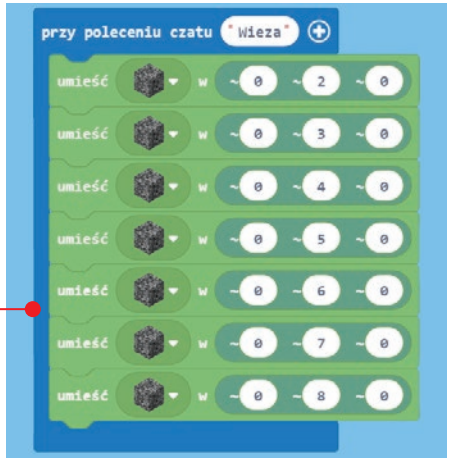



**2** Do wnętrza bloku dodajemy blok **umieść... w...**. Należy pamiętać, aby kliknąć na białą strzałkę, rozwinąć listę i wybrać z niej skałę macierzystą.

**3** W środkowym białym polu zamiast 0 wpisujemy **2**.

**4** Powtarzamy punkty **2** i **3**. Ważne, aby w każdym następnym bloku **umieść... w...** wpisywać liczbę o 1 większą od poprzedniej.

**5** Po uruchomieniu skryptu w świecie Mine crafta pojawi się stworzona przez nas struktura. Bardzo ważne jest, aby




w czasie wykonywania się kodu nie poruszać postacią – lokalizacja postaci jest punktem odniesienia. Dlatego jeśli na przykład poruszymy się w prawo po pojawieniu się drugiego bloku, to trzeci blok pojawi się na

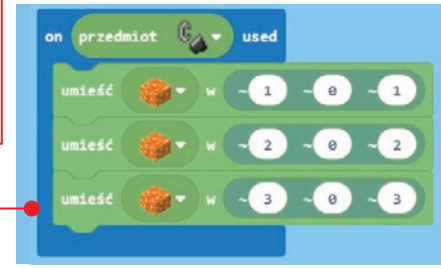
wysokości trzech bloków, ale również z przesunięciem w prawo. Zmiana lokalizacji postaci może spowodować zepsucie wieży.

# Kategoria Gracz

W blokach gracza znajdziemy elementy związane z czynnościami, jakie może podejmować gracz, takimi jak strzelanie, teleportacja czy wypowiadanie słów. Elementy te pozwalają je do czegoś zaprogramować.

**Przykład:**

Powyższy skrypt odpowiada za tworzenie deszczu lawy za każdym razem, gdy gracz wystrzeli strzałę.



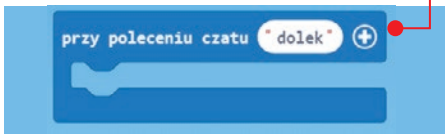

**Wypróbuj:**
A teraz wypróbujmy ten skrypt i sprawdźmy, co się wydarzy po jego wykonaniu.

# Kategoria Bloki

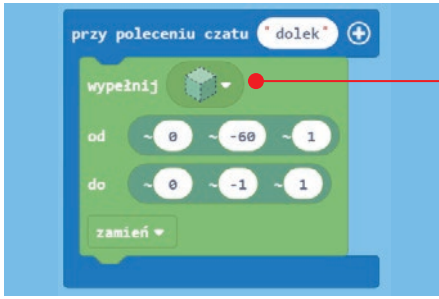
Kategoria ta zawiera wszelkie elementy związane z dodawaniem różnych bloków do otaczającego nas świata. Jeden z bloków **umieść... w...** jest zawarty w kategorii bloków podstawowych. A za pomocą całej tej kategorii można dowolnie kształtować teren.

**Przykład:**
Spróbujemy stworzyć olbrzymią dziurę w ziemi sięgającą aż granicy skały macierzystej.

**1** Aby to zrobić, należy z kategorii bloków gracza dodać blok komendy tekstowej.




**2** Następnie do wnętrza bloku komendy tekstowej dodajemy blok **wypełnij** i z listy rozwijanej wybieramy blok powietrza.

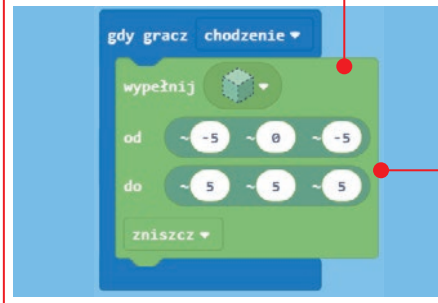



**3** Dzięki temu wszystkie bloki od skały macierzystej do pozycji gracza zostaną zastąpione powietrzem, tworząc dół. W efekcie otrzymamy wielki dół sięgający skały macierzystej.

## Miniprojekt Kopacz

Spróbujemy teraz stworzyć kopacza kopiącego tylko w kierunku poziomym. Wykorzystamy do tego również blok **wypełnij** – z jedną różnicą: kopanie będzie następowało przy każdym kroku bohatera w kierunku, w którym zwrócona jest jego postać. Potrzebujemy do tego w zasadzie tylko kilku bloków.

**1** Zaczynamy od bloku **gdy gracz <chodzenie>**, a następnie do jego wnętrza wstawiamy blok **wypełnij**.

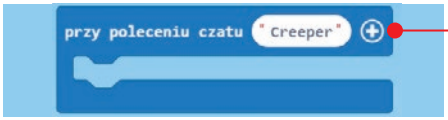



**2** W białych polach z liczbą 0 wpisujemy kolejno **-5**, **0**, **-5** w pierwszym rzędzie i **5**, **5**, **5** w drugim. Otrzymamy w ten sposób skrypt. Pozwoli nam on na stworzenie na przykład bardzo głębokiego tunelu.

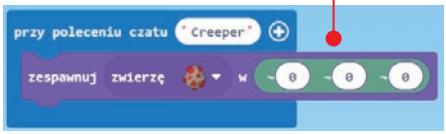
# Kategoria Moby

W tej kategorii znajdziemy bloki związane między innymi z dodawaniem lub usuwaniem mobów (wszelkiego rodzaju), dodawaniem różnych efektów do postaci graczy lub mobów (spowolnienie, przyspieszenie, obrażenia itp.). Bloki te składają się głównie z bloków strukturalnych i bloków uzupełniających. Oto najczęściej wykorzystywane bloki z kategorii moby.

**Przykład:**
Stworzymy skrypt sprawiający, że w danej lokalizacji pojawi się Creeper. Niestety, blok **zespawnuj... w...** pozwala na zespawnowania tylko zwierzęcia. Należy więc użyć bloku uzupełniającego z napisem **potwór** – to piąty blok z kategorii moby – i wstawić na miejsce bloku **zwierzę** w bloku **zespawnuj... w...**.




**1** Zaczynamy od bloku komendy tekstowej. Następnie wrzucamy do niego blok **zespawnuj... w...**.




**2** Potem blok **potwór** umieszczamy w miejsce bloku **zwierzę**. W efekcie otrzymamy skrypt przywołujący Creepera.

Dzięki naszemu skryptowi jesteśmy w stanie zespawnować Creepera w lokalizacji naszej postaci. **Uwaga!** Jeżeli jesteśmy w trybie Przetrwanie, to Creeper od razu nas zaatakuje, próbując wybuchnąć.

## Z RAPORTU STEVE'A

Zarówno blok **zwierzę**, jak i blok **potwór** ma rozwijalną listę. Po kliknięciu na nią zobaczymy wiele różnych jaj do wyboru – każde z nich odpowiedzialne jest za pojawienie się innego moba.

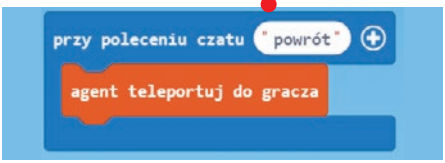
# Kategoria Agent

AGENT

Agent jest malutkim robotem, którego działanie można zaprogramować tak, jak tylko chcemy. Jest to przydatne. Na przykład gdy Agent będzie tworzył dla nas jakąś budowlę lub tunel, my będziemy mogli w tym czasie robić inne rzeczy.

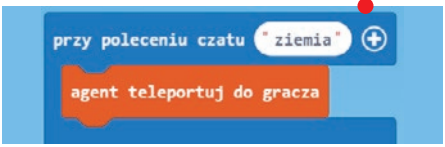
Nareszcie moja kategoria. W tej kategorii możesz zaprogramować moje działanie, abym tworzył dla ciebie skomplikowane struktury.

Agent na mapie wygląda w ten sposób. Warto za pomocą skryptu teleportacyjnego umieszczać go obok siebie po skończonej pracy – wtedy łatwo można wyznaczyć mu nowe lokalizacje do działania. A oto przykład skryptu teleportacyjnego.

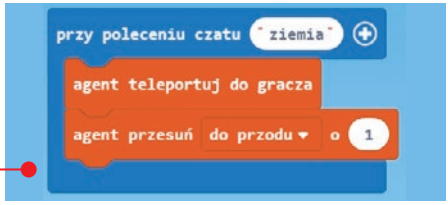



**Przykład:**
Zaprogramujmy Agenta do przygotowania ziemi pod prostą uprawę. Wykorzystamy do tego motykę.

**1** Zaczynamy oczywiście od teleportacji Agenta do naszej postaci.




**2** Po teleporcie zaczynamy od bloku przesuwającego Agenta o jedną pozycję do przodu.



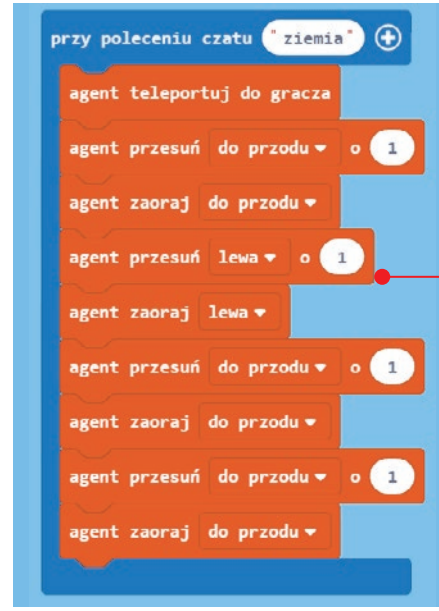

**3** Kolejnym etapem jest użycie bloku wykorzystującego motykę do orania ziemi, a tym samym do jej użyźniania.

**4** Etap z przesuwaniem i oraniem należy powtórzyć kilkakrotnie, aby mieć możliwość zwiększenia oranego pola. Im więcej dodamy bloków **agent przesuń** i **agent zaoraj**, tym dłuższe pole otrzymamy. W ten

## Z RAPORTU STEVE'A

Nie wszystkie bloki mogą być zamienione na pola uprawne. Upewnij się, że postać Agenta zostanie umieszczona na właściwym bloku.

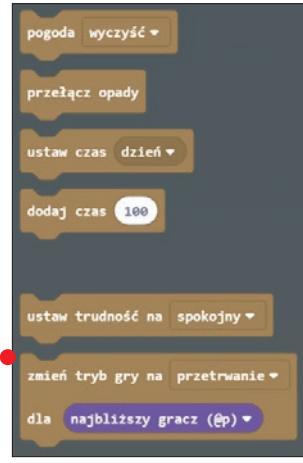
sposób możemy otrzymać długie, proste pole. Nie ma co czekać, trzeba coś na nim zasiać!




Oczywiście można zedytować trasę Agenta i stworzyć większe pole uprawne. Spróbujmy, klikając na rozwijalną listę (biała strzałka), wybrać inne opcje i poprowadzić Agenta w inne miejsca, aby stworzyć szersze lub bardziej skomplikowane pole.

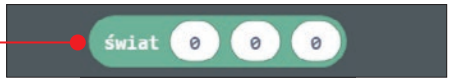
# Kategoria Rozgrywka

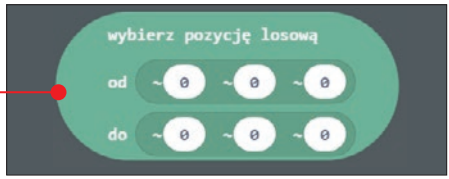
W tej kategorii nie znajdziemy zbyt wielu bloków. Są tu bloki w gruncie rzeczy odpowiedzialne jedynie za niektóre, choć bardzo ważne elementy gry. Najczęściej używane bloki to przede wszystkim zmiana poziomu trudności, trybu oraz manipulowanie pogodą i czasem. Na przykład ten skrypt odpowiedzialny jest za ustawienie na początku gry burzy o zmierzchu w trybie Przetrwanie. Z ustawień tych najczęściej korzysta się na początku gry, kreując wartości początkowe (startowe) rozgrywki.



# Kategoria Pozycje

To najmniejsza ze wszystkich kategorii. Ma tylko kilka bloków. Zawiera bloki pozwalające na edytowanie lokalizacji. Na przykład jeżeli chcemy, żeby dany element pojawił się w pozycji losowej, to jest to idealne miejsce na poszukiwanie bloku, który pozwoli to osiągnąć. Na największą uwagę zasługują w tej kategorii dwa bloki.

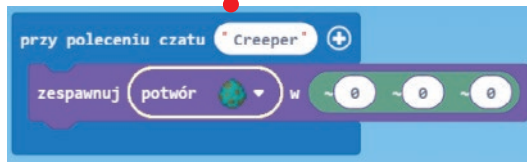


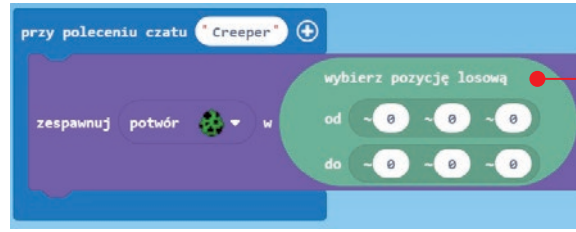




Pierwszy z nich pozwala na edycję lokalizacji wobec świata, a nie postaci, a drugi – na losowanie lokalizacji.

**Przykład:**
Na stronie 25 poznaliśmy skrypt Creepera. Wykorzystamy go teraz, aby nasz Creeper pojawiał sie w losowym miejscu. Powiedzmy, że Creeper rzucił nam wyzwanie. Oto co powinniśmy zrobić.

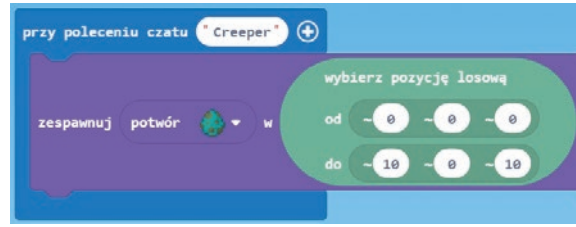
**1** Zaczynamy od wykorzystania bloku, który powodował pojawienie się Creepera w określonej pozycji.




**2** Następnie w miejsce trzech białych pól wkładamy blok **wybierz pozycję losową od... do...**.

**3** W białe pola wpisujemy liczby stanowiące zakres, w jakim ma się pojawić nasz Creeper. Wprowadźmy przykładowe wartości. Dzięki temu Creeper będzie losował sobie miejsce pojawienia się. Zakres to 10 bloków od pozycji gracza w każdą stronę na wysokości gracza.



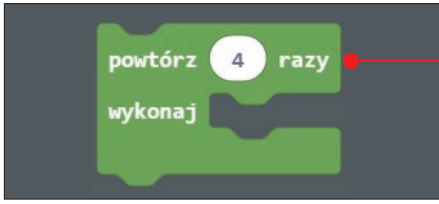
# Kategoria Pętle

Bardzo często chcemy, aby coś działało przez cały czas – od momentu uruchomienia programu do momentu jego zakończenia. Innym razem wolimy, by coś działało tylko przez określony czas albo do określonego momentu, w którym coś się wydarzy. Żeby to zrealizować, w programowaniu używamy pętli. Istnieje kilka rodzajów pętli – każda z nich ma swoje charakterystyczne właściwości. Na platformie MakeCode również znajdziemy kilka rodzajów pętli.

## Pętla ciągła

To najczęściej wykorzystywany rodzaj pętli. Pozwala na powtarzanie bloków, które znajdą się w jej wnętrzu, od momentu rozpoczęcia programu do jego zakończenia. Dzięki temu dana czynność jest w zasadzie wykonywana zawsze. Przykładem może być cykl dnia i nocy w Minecrafcie – proces ten odbywa się ciągle od startu gry do jej wyłączenia.

## Pętla ograniczona liczbą powtórzeń

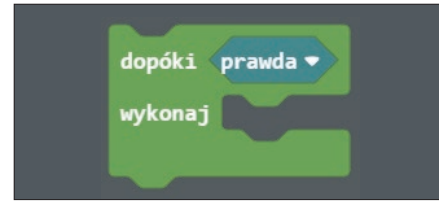



Ten rodzaj pętli sprawia, że dana funkcja wykona się tylko określoną liczbę razy. Wszystko zależy od liczby, jaka zostanie wpisana w białe pole. Na przykład w przypadku pętli wszystkie bloki wewnątrz pętli wykonają się cztery razy.

## Z RAPORTU STEVE'A

- Cykl dnia i nocy trwa w grze 20 minut
- Dzień trwa 10 minut
- Noc trwa 7 minut
- Zachód i wschód mają po 1,5 minuty

## Pętla ograniczona warunkiem



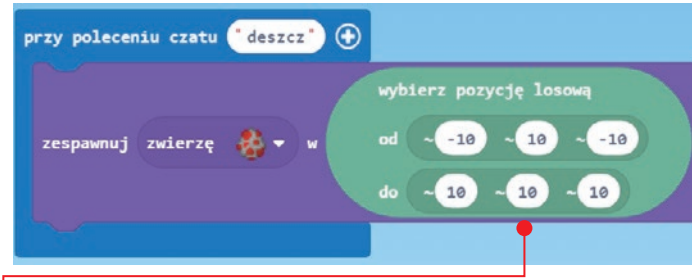

Pętla ograniczona warunkiem to specyficzny rodzaj pętli. Dana czynność wykonuje się, dopóki jest spełniany jakiś warunek. Gdy warunek przestanie być spełniany, wszystko, co się znajduje wewnątrz tej pętli, zatrzyma się. Najlepszym przykładem na działanie pętli ograniczonej warunkiem jest zachowanie Creepera. Nasz dobry przyjaciel wędruje sobie bez celu po mapie. Jest to więc czynność powtarzana. Warunkiem, który tutaj działa, jest brak gracza w okolicy. Gdy gracz pojawi się w bliskim otoczeniu (warunek przestaje być spełniany), wtedy Creeper przerywa powtarzaną czynność i śledzi gracza, mając na celu wybuch.

## Pauza

Poza pętlami wartym uwagi blokiem z tej kategorii jest bloczek pauzy, który pozwala na robienie przerw w czynności zaprogramowanej postaci.

**Przykład:**
Spróbujemy wykorzystać pętle do stworzenia deszczu kurczaków. Zwykle z nieba spada na nas deszcz, grad czy śnieg. Może być jednak inaczej – z nieba będą spadać kurczaki!

**1** Zaczynamy od znanego nam już skryptu związanego z pojawianiem się postaci. W tym przypadku jest to kurczak, który pojawiać się będzie na wysokości 10 bloków w odległości 10 bloków od gracza w każdą stronę. Ten skrypt jednak spowodowałby pojawienie się tylko jednego kurczaka za każdym razem, gdy w konsoli czatu wpiszemy komendę deszcz.




**2** Aby sprawić, by dana czynność się powtarzała, do wnętrza bloku z poleceniem czatu wrzucamy pętlę, a do wnętrza tej pętli wprowadzamy istniejące już bloki **zespawnuj zwierzę**. Powinniśmy otrzymać skrypt.

**3** Po uruchomieniu skryptu otrzymamy deszcz kurczaków.

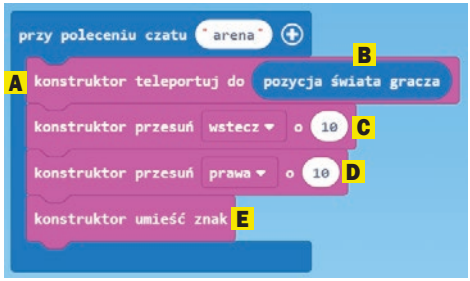
**Najwyższa pora wykorzystać zdobytą wiedzę i spróbować stworzyć własne projekty. Sprawdzimy twoją gotowość do zawładnięcia światem Minecrafta za pomocą bloków i skryptów.**
**Czekają cię dwa wyzwania – Arena i Parkour. Każde wyzwanie postaram się wyjątkowo dobrze opisać, aby nie było żadnych problemów.**
**Powodzenia!**

# Projekt Arena

Dawno temu w starożytnym Rzymie na arenie można było spotkać walecznych gladiatorów, którzy stawali do walki z innymi gladiatorami lub dzikimi bestiami. Dziś Steve będzie musiał stawić czoło mobom z Minecrafta i spróbować utrzymać się jak najdłużej na arenie. Po kolei pojawiać się będą kolejne moby, a poziom trudności będzie wzrastał. Pojedynki będą się odbywać na arenie zbudowanej ze skały macierzystej. Ważnym elementem będzie też uzbrojenie Steve'a, aby mógł walczyć.

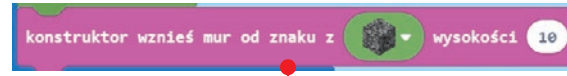
## Pierwsza część skryptu

**1** Zaczynamy od przygotowania konstruktora. Jest to blok z kategorii bloków zaawansowanych, działaniem bardzo przypomina Agenta. Pierwszy blok teleportuje

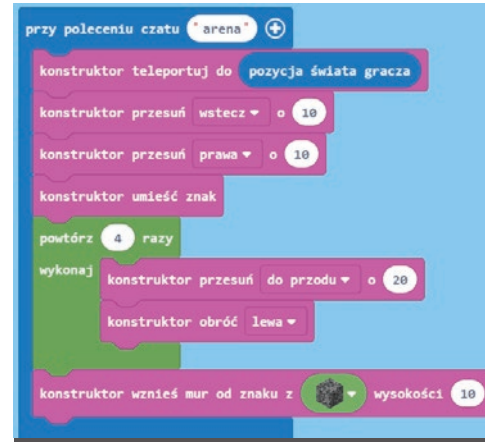



konstruktora do gracza **A**. Blok **pozycja świata gracza** **B** znajdziemy w kategorii **Gracz** i wstawiamy go zamiast trzech białych pól z liczbą 0. Potem konstruktor cofa się o **10** bloków od gracza **C** i idzie w prawo o **10** bloków **D**. Następnie umieszcza znak **E** (będzie to róg naszej areny). Znak nie jest widoczny i służy tylko jako punkt orientacyjny dla konstruktora, w którym miejscu ma zacząć budować.

**2** Kolejną czynnością jest zaplanowanie trasy konstruktora. Wędruje on po świecie, rysując kwadrat o długości boku 20 bloków **F**. Nie widzimy tego rysowania, bo dzieje się to w programie. Użycie pętli wykonującej się czterokrotnie **G** jest konieczne, by wykonać cztery ściany.




**3** Na koniec dodajemy wzniesienie muru. Dzięki temu na narysowanym planie areny pojawi się mur o wysokości 10 bloków.

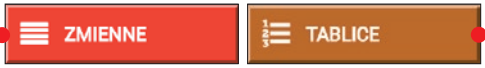



Cały skrypt pierwszej części

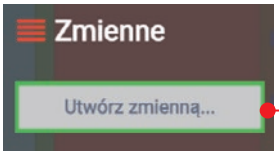
W drugiej części zastosujemy tablice. Tablica jest to rodzaj listy, z której gra będzie wybierać przeciwnika.

## Druga część skryptu

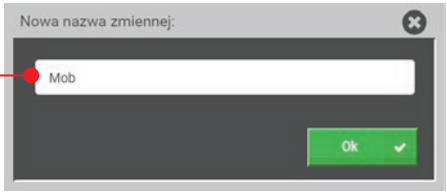
Zastosujemy w drugiej części zmienne oraz tablice. Zmienną w naszym



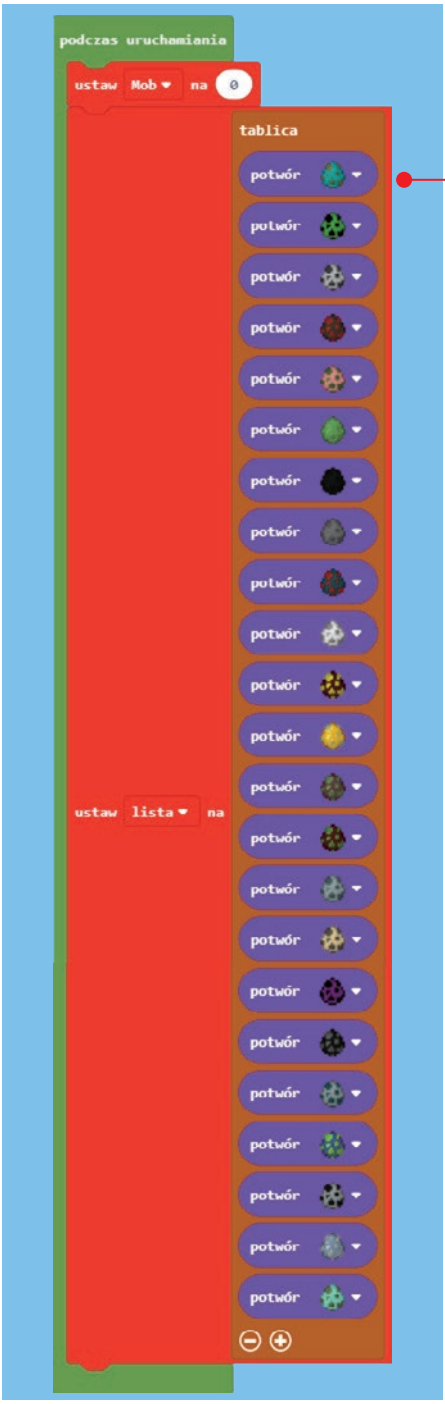

przypadku będzie przeciwnik (będzie się zmieniał co jakiś czas). Jednak aby gra mogła wybierać różnych przeciwników, musimy jej stworzyć listę postaci, z którymi będziemy walczyć (z tej listy będzie wybierany przeciwnik). Tak jak nauczyciel wyczytuje z listy uczniów podczas sprawdzania obecności, podobnie nasz program będzie wyczytywał moby z listy mobów.




**1** Zaczynamy od dodania zmiennej (klikamy na ●, a potem na ●) oraz nadania jej nazwy **Mob** ●. Zmienna to wartość, która się zmienia w trakcie działania programu. U nas mob będzie się zmieniał co pewien czas, tak aby mieć możliwość walki z wszystkimi przeciwnikami.




**2** Tworzymy tablicę, klikając na ●, na której wylistowane będą wszystkie dostępne agresywne moby. Po kliknięciu na plus ●

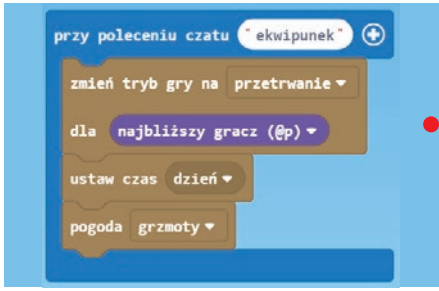


pojawi się puste pole. W pole to wrzucamy blok **potwór** z kategorii **Moby**, tworząc listę. Program po kolei będzie sprawdzał przeciwników i wrzucał ich nam na arenę. Ostatecznie skrypt będzie wyglądał następująco.

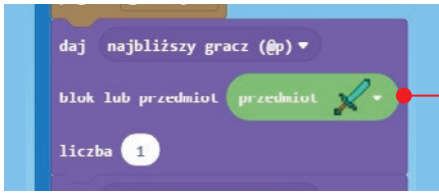
## Trzecia część skryptu

W trzeciej części wręczymy Steve'owi ekwipunek początkowy, aby miał możliwość walki w pełnym uzbrojeniu i orężu.

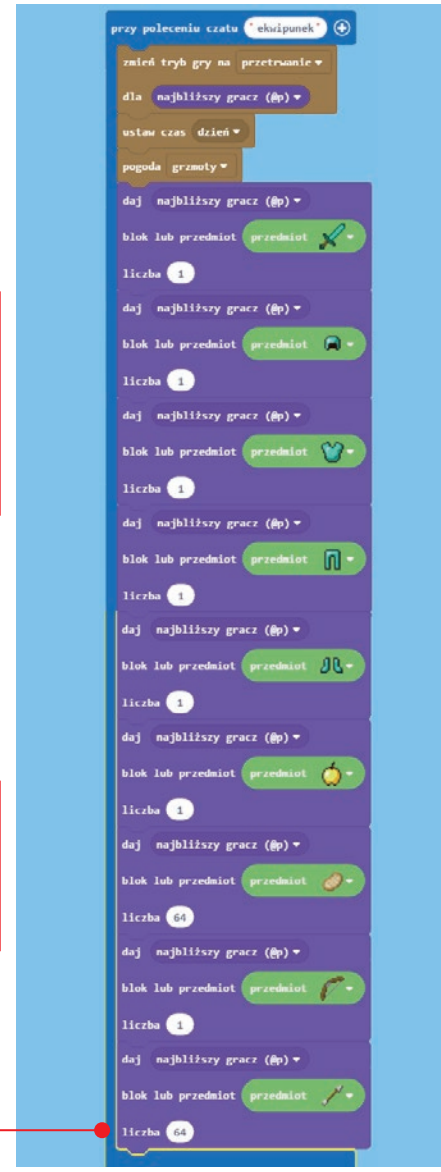
**1** Pierwszy krokiem będzie ustawienie rozgrywki, czyli zmiana trybu gry na Prze-




trwanie, ustawienie pory dnia na dzień oraz dodanie burzy jako pogody początkowej. Dzięki temu nadamy specjalny klimat całej rozgrywce. Dodanie burzy ma jeszcze jeden aspekt – deszcz sprawi, że moby nieodporne na słońce nie będą dostawać od niego obrażeń.

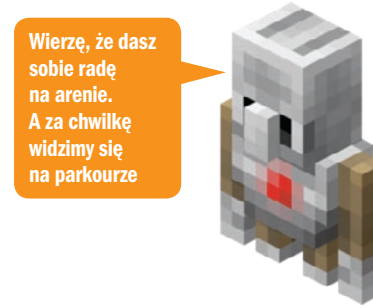



**2** Do obecnego skryptu dodajemy blok z kategorii **Moby** i uzupełniamy go blokiem **przedmiot** z kategorii **Gracz**. Dzięki temu po wpisaniu komendy **ekwipunek** otrzymamy diamentowy miecz. Blok ten możemy dodawać wielokrotnie, tak aby dać graczowi pełne uzbrojenie. Propozycje uzbrojenia i oręża znajdziemy w pełnym skrypcie trzeciej części.



## Czwarta część skryptu

Ostatnia (i najkrótsza) część skryptu to zaprogramowanie wybierania kolejnych mobów na arenę. Chodzi o to, by sprawić, żeby gra wybierała po kolei z tablicy moby i wrzucała je na arenę. Między kolejnymi „wrzuceniami" mobów powinna być przerwa, na przykład 15 000 ms (inaczej mówiąc, 15 sekund). Jeżeli nie uda nam się w tym czasie pokonać przeciwnika, pojawi się kolejny i trzeba będzie walczyć z dwoma naraz. Każdą komendę warto uruchamiać pojedynczo, aby przygotować wszystko na kolejną turę. Pamiętajmy również, że komenda **walka** jest komendą ostateczną i po jej aktywowaniu nie można już zatrzymać rozgrywki – trzeba walczyć z mobami.

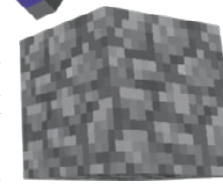


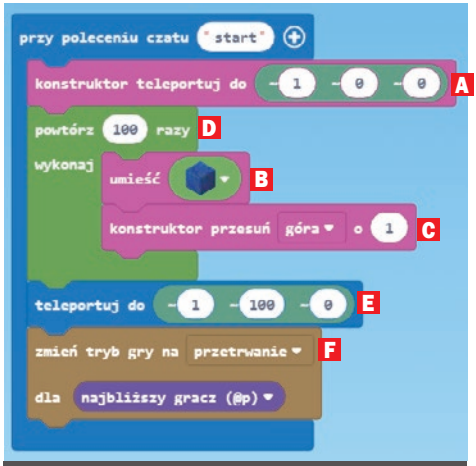

# Projekt Parkour

Parkour to forma sportu polegająca na pokonywaniu różnych przeszkód jak najprostszym sposobem poprzez wykonywanie akrobacji, skoków itp. W Minecrafcie można znaleźć wiele różnych map i serwerów opartych tylko na parkourach. Zobaczmy, jak samemu stworzyć prosty parkour.

## Etap 1

1 Na początku przenosimy konstruktora obok pozycji gracza **A**. Następnie stawiany jest blok lazurytu **B** i konstruktor przenosi się o jeden blok w górę **C**. Czynność ta powtarzana jest **100** razy **D**. W tym czasie konstruktor przerywa pracę, a nasza postać teleportowana jest na samą górę wieży **E**. Ostatnim etapem jest zmiana trybu gry na





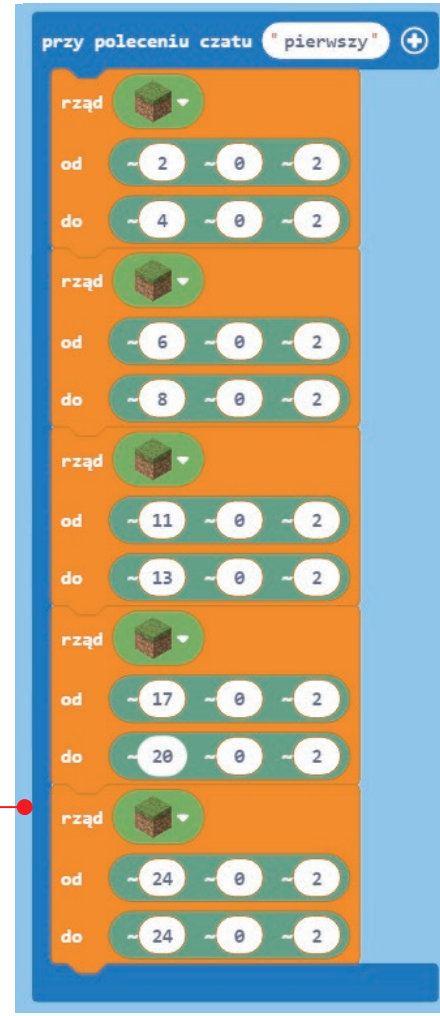

**Skrypt odpowiedzialny za stworzenie wielkiej wieży za pomocą konstruktora**

**przetrwanie** F. Od tej pory gra już się rozpoczyna.

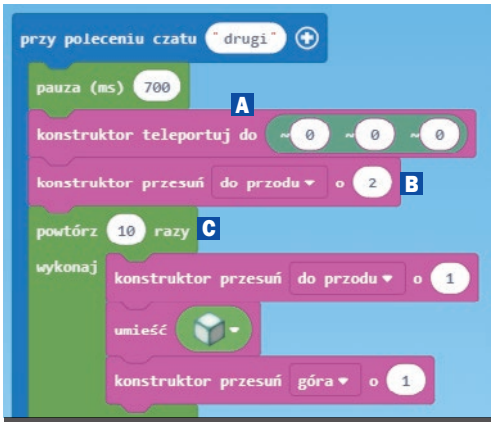
**2** Teraz skorzystamy z kategorii **Kształty**.

Jest to kategoria bloków zaawansowanych, jednak służy głównie ułatwianiu tworzenia kształtów geometrycznych. Stworzymy pięć rzędów zbudowanych z bloków ziemi porośniętych trawą, które będą znajdować się w pewnej odległości od siebie nawzajem. Naszym zadaniem będzie przeskakiwanie z bloków na bloki, unikając upadku.

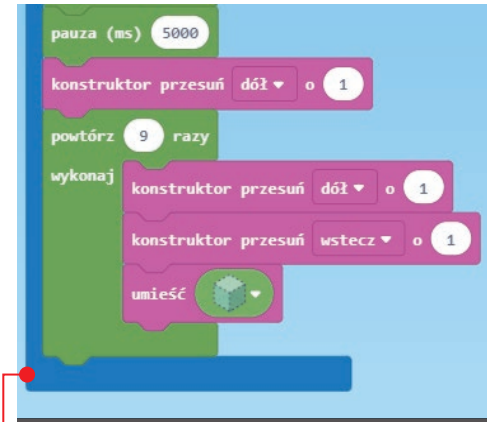
W kształtach znajdziemy również kulę, która zbuduje się dla nas z dowolnie wybranego materiału. Jedyne, co musimy zrobić, to określić, gdzie będzie się znajdować środek kuli oraz jak duża ona będzie.




**Każdy z etapów programowania wypełniaj po przejściu w grze etapu Parkouru. Nie uruchamiaj wszystkich skryptów naraz!**




**Skrypt odpowiedzialny za stworzenie schodów z latarni morskiej (blok)**




**Skrypt sprawiający, że schody zamieniają się w powietrze, a tym samym znikają**

## Etap 2

**1** Najpierw konstruktor teleportuje się do aktualnej pozycji gracza **A**, a następnie przesuwa się o dwa pola do przodu **B**.

**2** Kolejnym etapem jest pętla, która wykonuje się **10** razy **C** – jej zadaniem jest zbudowanie schodów.

**Uwaga!** Należy pamiętać, żeby nie uruchamiać na tym poziomie kodu, ponieważ zaraz zostanie dodany do skryptu kod odwrócony, który sprawi, że schody znikną.

## Etap 3

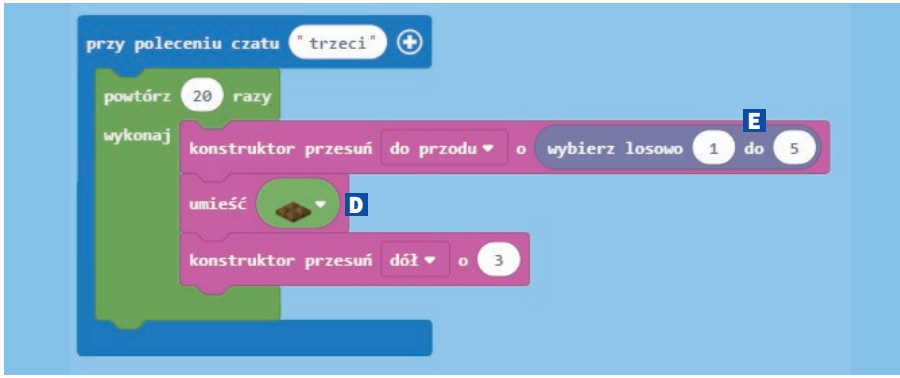
Dodajemy skrypt sprawiający, że schody zamieniają się w powietrze, czyli znikają. Należy pamiętać, aby po wywołaniu kodu za pomocą konsoli tekstowej od razu rozpocząć wspinaczkę po schodach ze względu na to, że bardzo szybko znikną.

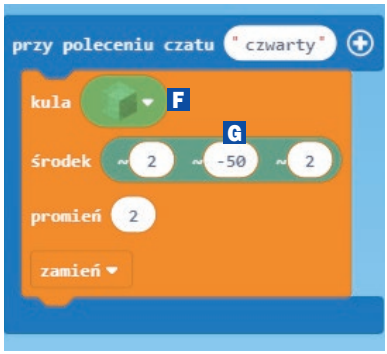
## Etap 4

Kolejny etap Parkouru pozwoli nam na podróż w dół – za pomocą klap podłogowych **D**. Konstruktor przejdzie w dół i losowo co 1, 2, 3, 4 lub 5 **E** bloków będzie umieszczać w świecie gry klapę podłogową. Naszym zadaniem będzie zeskakiwać po nich w dół.

## Etap 5

**1** Ostatni etap to finałowa kula szlamu **F**, na którą należy zeskoczyć. Kulę dodajemy za pomocą kategorii **Kształty**.

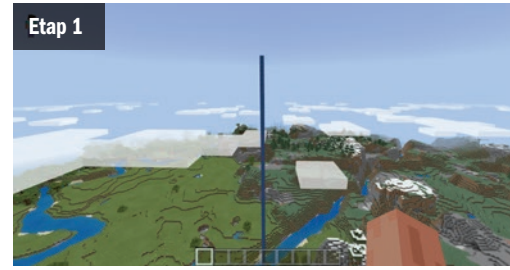


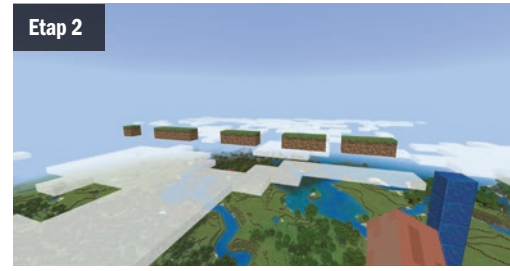



**2** Kula znajduje się **50** bloków pod bohaterem G. Należy więc uważać, bo jeśli zamiast na nią zeskoczymy na ziemię, będzie skutkować przegraną.

Jak ostatecznie będzie wyglądał nasz parkour, pokazują ilustracje obok (Etap 1 – Etap 5).

**Bardzo miło było cię poznać. Mam nadzieję, że się jeszcze kiedyś spotkamy. W dalszej części książki dowiesz się, jak tworzyć mody do Minecrafta oraz jak zmieniać świat Minecrafta z wykorzystaniem języka Python. Z pewnością poznasz wielu nowych przyjaciół, a także zmierzysz się z innymi mobami. Wpadaj czasem do mnie – do MakeCode – może wspólnie stworzymy jeszcze jakąś wieżę albo bunkier. Powodzenia w dalszej wędrówce!**



# 3 MCreator: mody do Minecrafta

**W tym rozdziale poznamy narzędzie MCreator, czyli platformę do tworzenia modów do Minecrafta. Przeczytamy też, jak w MCreatorze edytować biomy, programować moby i modyfikować rozgrywkę**

Co to są mody, czyli modyfikacje gier, wyjaśnia obok specjalista z wioski Minecrafta.

Zdarza się, że niektórzy producenci gier dodają do swoich gier edytory pozwalające na tworzenie modów. Czasami dotyczy to tylko map i światów, a czasami również i całej rozgrywki.

**Witaj. Jestem jednym z osadników. Opowiem ci, co to są mody i jak je wykorzystywać. Mody to inaczej modyfikacje już istniejących gier. Pozwalają na zmianę różnych elementów rozgrywki, dodawanie postaci i przedmiotów. Najczęściej mody tworzone są przez społeczność graczy, rzadziej przez producentów gier.**

Zdarza się też, że gracze samodzielnie łamią kod gry, aby tworzyć własne tryby rozgrywek. Tak się dzieje na przykład z serią gier Heroes of Might and Magic, a konkretnie z jej najpopularniejszą, ponadczasową odsłoną Heroes III. Można powiedzieć, że brak jakichkolwiek działań ze strony producentów gry sprawił, że fani wzięli sprawy we własne ręce. Doprowadziło to do powstania dwóch największych modyfikacji: Heroes III WoG (Heroes III: In the Wake of Gods) oraz Heroes III HotA (Heroes III: Horn of the Abyss). Obie modyfikacje uznawane są za klasykę wszelkich modyfikacji do gier i zajmują topowe miejsca w ogólnych rankingach modyfikacji.

Minecraft również ma wiele modyfikacji. Należy do jednej z najczęściej modyfikowanych gier na świecie. Ponadto istnieje możliwość korzystania z wielu modów jednocześnie w danej rozgrywce. Stąd też w internecie można znaleźć tysiące różnych modyfikacji. Niektóre z nich są na tyle popularne, że Mojang Studios bierze pod uwagę uwzględnienie ich w dalszych aktualizacjach świata Minecrafta.

## PROGRAMY DO TWORZENIA MODÓW ISTNIEJĄCYCH GIER

Najbardziej znane programy do tworzenia modów istniejących gier:

- **REDKit** do gry Wiedźmin 2: Zabójcy królów
- **Source SDK** do gry Half-Life 2
- **Mod Development Hit**do serii gier Gothic

Heroes III: In the Wake of Gods zyskało uznanie milionów graczy

Przykład moda dodającego kilkanaście nowych młotów, które potrafią rozbić 3x3 bloki w miejscu uderzenia

Mod pozwalający na widzenie przez bloki i lokalizowanie wartościowych rud surowców

Mod dodający równoległy do Netheru wymiar z nowymi blokami, przeciwnikami i obiektami

Każdy z modów dodaje coś nowego lub modyfikuje obecnie istniejące funkcje czy przedmioty. Aby poznać środowisko minecraftowych modyfikacji, dobrze jest przyjrzeć się kilku z nich, by wiedzieć, jakie elementy wprowadzają lub zmieniają.

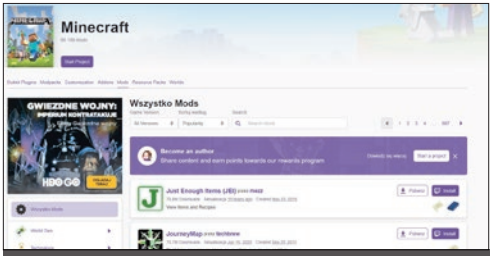
Bardzo trudno określić liczbę modów do Minecrafta. Wciąż pojawiają się dziesiątki nowych. Warto przeglądać strony internetowe katalogujące modyfikacje do różnych gier (w tym do Minecrafta) oraz korzystać z for internetowych dla graczy. Na wielu z nich znajdziemy nie tylko modyfikacje, ale też instrukcje ich instalacji oraz tutoriale, jak tworzyć własne mody za pomocą różnych programów i aplikacji.

Pixelmon łączy świat Pokemonów ze światem Minecrafta

Iron Chest wprowadza do świata nowe rodzaje skrzyń

### CurseForge: curseforge.com

To strona, na której znajdziemy większość modyfikacji z podziałem na kategorie i wprowadzane funkcje. Strona ta nie tylko pozwala na pobieranie modów, ale ma również wiele dodatkowych opcji, umożliwia na przykład dodawanie własnych modyfikacji. Znajdziemy na niej wiele fanowskich modów.

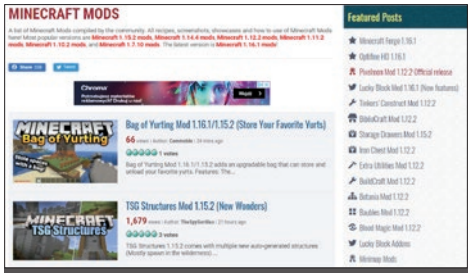



Strona CurseForge – curseforge.com

Mod zmieniający wygląd świata Minecrafta

### 9Minecraft: 9minecraft.net

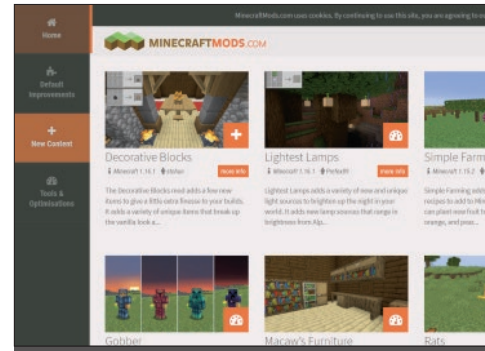
Strona z różnymi modyfikacjami, która umożliwia także recenzowanie pobranych modów oraz wystawianie im ocen, dzięki czemu jesteśmy w stanie sprawdzić popularność różnych modów i wybrać te najbardziej sprawdzone.




Strona 9Minecraft – 9minecraft.net

### MinecraftMods.com

Strona ze sprawdzonymi modami. Jej zaletą jest duża przejrzystość i profesjonalne opisy każdej modyfikacji.



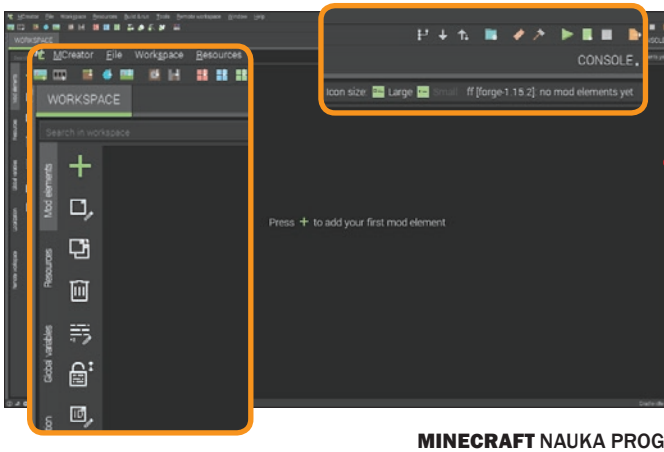

Strona MinecraftMods.com – minecraftmods.com

# Programy do tworzenia minecraftowych modów

Popularność różnego rodzaju modyfikacji Minecrafta sprawiła, że pojawiło się wiele różnych programów do ich tworzenia. W dalszej części książki skupimy się na **MCreatorze** – to jedna z najlepszych opcji, jednak warto poznać również inne programy (patrz kolejna strona).

## Z RAPORTU STEVE'A

**Tekstura to wypełnienie powierzchni obiektu graficznego wzorem lub fakturą, na przykład wzór cegieł na murze, przezroczysta, pofalowana tafla wody na jeziorze.**

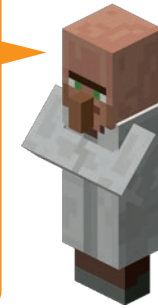



### MCreator

Najpopularniejszy program wykorzystywany do tworzenia modów i tekstur w Minecrafcie. MCreator jest jednym z najwygodniejszych programów ze względu na łatwą obsługę i bardzo intuicyjne rozmieszczenie wszelkich elementów. Pozwala na stworzenie wielu cie-

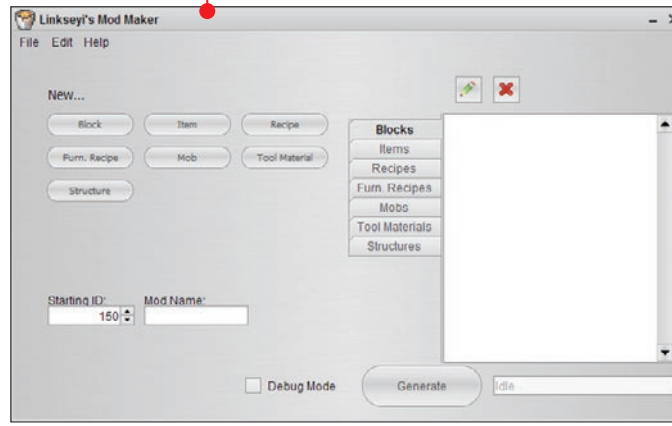
kawych modów, jednak żeby tworzyć od podstaw bardziej skomplikowane, nowe modyfikacje, trzeba korzystać z kodu źródłowego i manipulować nim w odpowiednich edytorach.
MCreator ze względu na swój przejrzysty interfejs i łatwość użytkowania jest bardzo chętnie wykorzystywany przez graczy.

**Pamiętaj, aby nie pobierać i nie instalować programów niewiadomego pochodzenia. Na tej stronie są wymienione najpopularniejsze programy do tworzenia modów. Jeżeli znajdziesz w internecie inne, oferujące dużo więcej funkcji, to upewnij się dokładnie, czy nie są to jakieś programy szpiegowskie lub wirusy!**



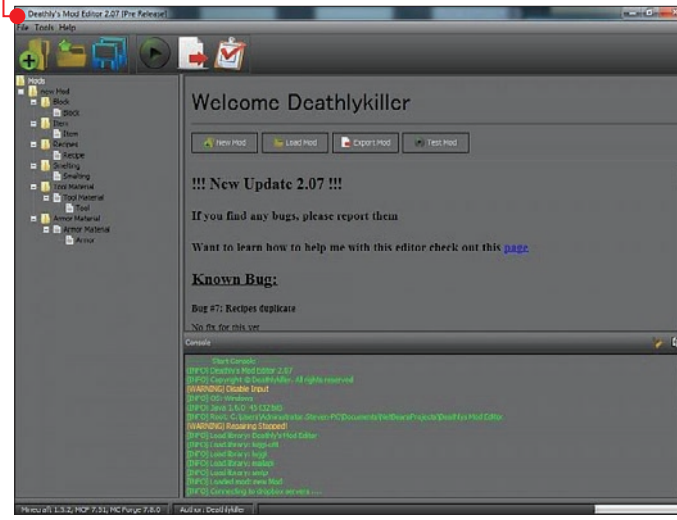
## Linkseyi's ModMaker Tool

Linkseyi's ModMaker Tool ma dużo możliwości, jednak jest o wiele mniej popularny. Pozwala na kreowanie własnych mobów, itemów, bloków i biomów. Wszystko w programie jest przedstawione w prosty sposób.




## Deathly's Mod Editor

Deathly's Mod Editor to edytor modów mający podobne możliwości co Linkseyi's ModMaker Tool. Niestety, toporny interfejs, kolorystyka i nieintuicyjne rozmieszczenie elementów i funkcji sprawiają, że nie jest on popularnym rozwiązaniem.




Wymienione trzy edytory modów są uważane za najlepsze. Istnieje też wiele innych podobnych aplikacji i ich przeróbek, które oferują podobne (często nawet takie same) funkcje, a różnią się w zasadzie tylko nazwami. W tym rozdziale będziemy poznawać MCreatora ze względu na łatwość użytkowania i mnogość możliwości.

# Skąd wziąć MCreatora

Najwyższa pora rozpocząć pracę z MCreatorem. Czeka nas wiele pracy związanej z modami i teksturami, ale aby rozpocząć, musimy pobrać i zainstalować oprogramowanie.

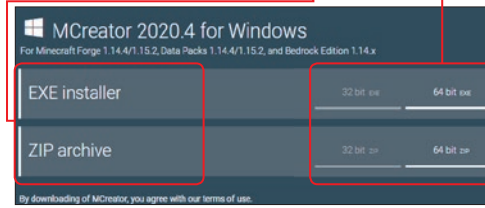
**1** **MCreatora** (**DVD-KOD: 002/003 32-/64-BIT**) znajdziemy na płycie dołączonej do książki. Można go też pobrać ze strony producenta. W tym celu w przeglądarce wpisujemy adres **mcreator.net**. Na stronie tej znajdziemy nie tylko możliwość pobrania MCreatora, ale również inne przydatne opcje:

- **Community** – tu możemy oglądać projekty innych użytkowników,
- **Forum** – tu możemy zadawać pytania, podzielić się wątpliwościami lub podyskutować na tematy związane z modyfikacjami,
- **Lista najpopularniejszych modów** – to może być źródło inspiracji, gdy zaczniemy tworzyć własne modyfikacje.

**2** Aby pobrać program ze strony producenta, klikamy na niej na **Download MCreator**. Widzimy, że program jest dostępny na różne systemy.

**3** Zostaniemy przeniesieni na zakładkę, na której znajdziemy najbardziej podstawowe informacje, takie jak wymagania systemowe, instrukcję pobrania itp. (w języku angielskim). Klikamy na przycisk **Download** obok nazwy programu.

**4** Zostajemy przeniesieni do wyboru pliku. Wybieramy instalator odpowiedni dla naszego systemu operacyjnego – w naszym poradniku będzie to Windows, w zależności od wersji systemu wybieramy 32 bit lub 64 bit. Do wyboru mamy dwa pliki.




- **EXE installer** – jest to plik instalacyjny
- **ZIP archive** – spakowany plik instalacyjny, który po pobraniu należy rozpakować przy użyciu dowolnego programu do rozpakowywania i pakowania plików.

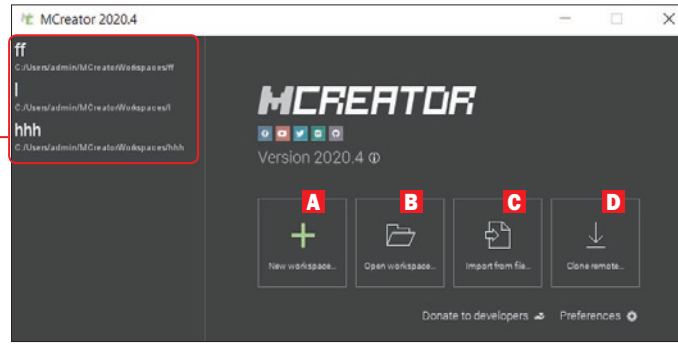
**5** Wybieramy plik, który nam odpowiada, pobieramy na dysk i rozpoczynamy instalację. Proces ten przebiega standardowo – wystarczy kliknąć na **Next** w pierwszym oknie instalatora, zaakceptować licencję i kliknąć na **Install**.

# Rozpoczęcie pracy z MCreatorem

## Okno programu

Po uruchomieniu programu na początku zobaczymy okno startowe. Po lewej stronie przy kolejnych uruchomieniach programu będzie widać nasze ostatnie projekty.

A w środkowej części okna są przyciski:




- **A** Rozpoczęcie nowego, pustego projektu.
- **B** Otworzenie już istniejącego projektu.
- **C** Import gotowej modyfikacji, którą można edytować.
- **D** Pobranie kopii aktualnej wersji projektu ze zdalnego repozytorium, jak GitHub.

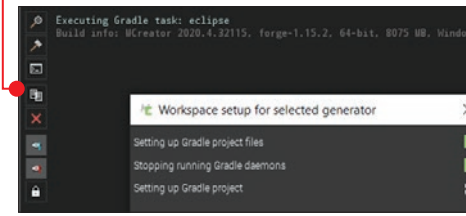
## Tworzymy nowy projekt

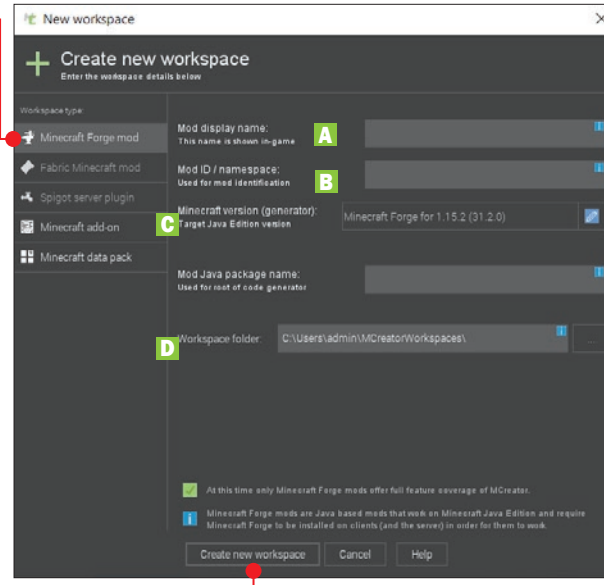
Po kliknięciu na **A** określamy szczegóły nowego projektu.

Po lewej stronie wybieramy rodzaj moda, w naszym przykładzie – **Minecraft Forge mod**, to dobry wybór na początek ze względu na to, że pozwala łatwo stworzyć modyfikację. Kolejne pola to:

- **A** **Mod Display name** – tu wpisujemy nazwę moda.
- **B** **Mod ID** – to miejsce na nazwę modyfikacji w plikach systemowych gry. Na początkowych etapach pracy z modyfikacjami nic tu nie wpisujemy.
- **C** **Minecraft version (generator)** – wersja gry, w której tworzony jest dany mod.
- **D** **Workspace folder** – miejsce, w którym będzie zapisany nasz projekt.

Po kliknięciu na **Create new workplace** zostajemy przeniesieni do ekranu, w którym nastąpi pobranie potrzebnych plików i przygotowanie niezbędnych elementów.



WORKSPACE

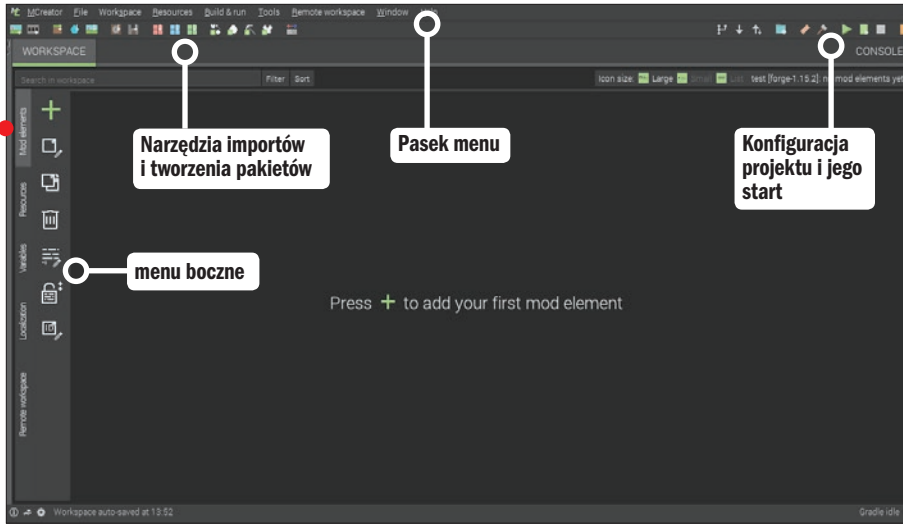
Po zakończonym procesie klikamy na napis **Workspace**. Jesteśmy już gotowi, by rozpocząć pracę – pokazuje się okno edycji. Program jest w języku angielskim, ale kolejne wskazówki zapoznają nas krok po kroku z podstawami tworzenia własnych modyfikacji do Minecrafta.

Aby poznać możliwości MCreatora, rozpoczniemy naszą pracę od stworzenia bardzo prostego bloku, który zaimplementujemy do gry w postaci moda.

## Z RAPORTU STEVE'A

Modyfikacje można tworzyć nie tylko do gier, ale również do całych serwerów. Są to pluginy, które wgrywa się po stronie serwera. Mogą one dodawać wiele różnych funkcji nie tylko urozmaicających rozgrywkę, ale też zabezpieczających grę przed trollingiem czy przed grieferami (graczami celowo psującymi rozgrywkę).

**Już za moment rozpoczniesz pracę z tworzeniem modyfikacji. Musisz pamiętać o tym, aby twój mod nigdy nie służył utrudnianiu innym rozgrywki poprzez griefing, trolling czy inne elementy krzywdzące społeczność graczy. Nie bądź Creeperem dla innych graczy!**




# Miniprojekt Blok piór

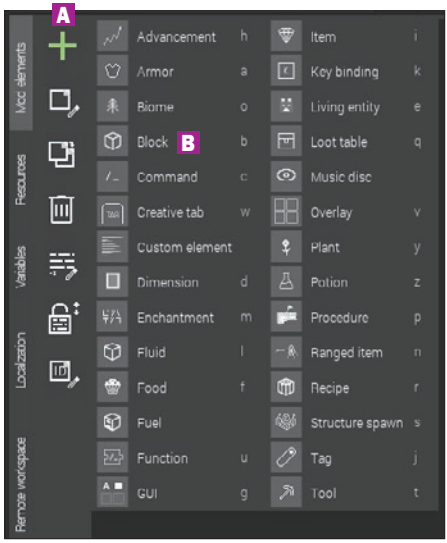
Pióra to przedmiot upuszczany przez papugi i kurczaki. Są kluczowe na wstępnych etapach gry, ponieważ pozwalają stworzyć strzały,

które są niezbędne, by móc korzystać z łuku. Innym zastosowaniem piór jest stworzenie książki z piórem oraz gwiazdki pirotechnicznej.

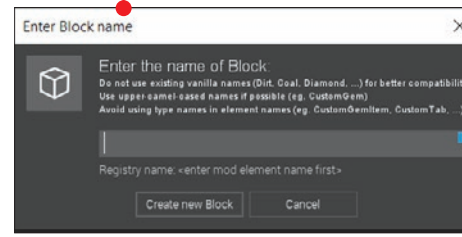
## Z RAPORTU STEVE'A

**W Minecrafcie istnieje 29 000 000 000 000 000 000 (29 trylionów) różnych rodzajów gwiazdek pirotechnicznych. Nadają one kolor fajerwerkom.**

Pióra nie występują w formie bloków. Naszym zadaniem będzie stworzenie takiego bloku, który po zniszczeniu będzie wyrzucał dziewięć piór.



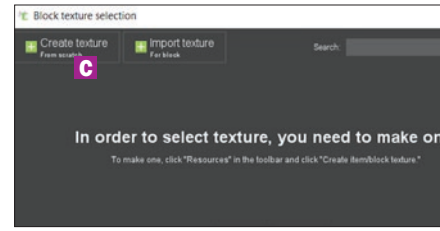

**1** Klikamy na plus **A** i wybieramy opcję **Block B**. Zostajemy poproszeni o nazwanie bloku ●. Klikamy na **Create new Block**.

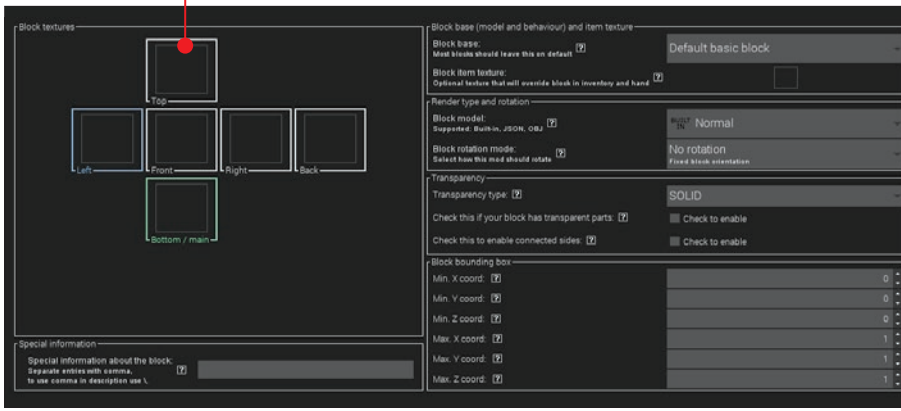



**UWAGA!**

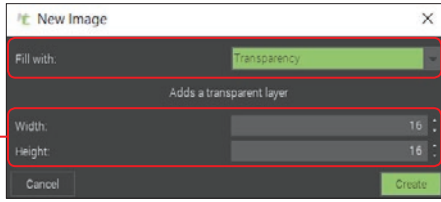
**W nazwach projektów, bloków, itemów nie używamy spacji, polskich znaków i znaków specjalnych, takich jak % ^ #.**

**2** Zostajemy przeniesieni do okna ●, w którym możemy zaprojektować blok. Zaczniemy od tekstury. Klikając na dowolny kwadrat ● po lewej stronie, wywołamy okno tworzenia tekstury.



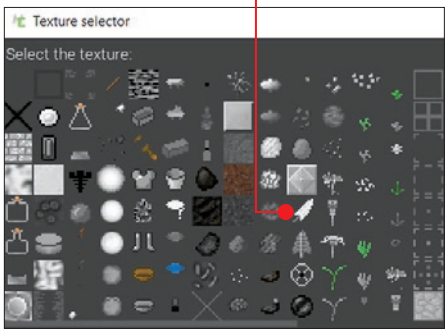



**3** Klikamy na przycisk **Create texture** C – powoduje uruchomienie kreatora tworzenia tekstury.



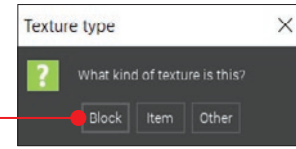

**4** W nowo otwartym oknie wybieramy długość i szerokość (bazowo jest to 16 pikseli). W sekcji **Fill with:** są do wyboru cztery opcje: **Transparency**, **Color**, **Template**, **No layer**.

**5** Wybieramy **Template** i klikamy na kwadrat, który znajduje się pod tą opcją. Otwiera się okno, w którym możemy wskazać gotowy element tekstury dla naszego bloku.

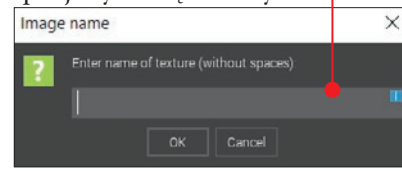



**6** Nowe okno to narzędzie do malowania i rysowania podobne do programu **Paint**. Możemy spróbować samodzielnie namalować teksturę bloku, czyli to, jak będzie wyglądała jego ściana. My jednak, dzięki skorzystaniu z opcji **Template**, mamy już gotowe pióro, które możemy edytować. Wykorzystujemy do tego różne narzędzia po lewej stronie, takie jak ołówek, pędzel, dodawanie kształtów itp.

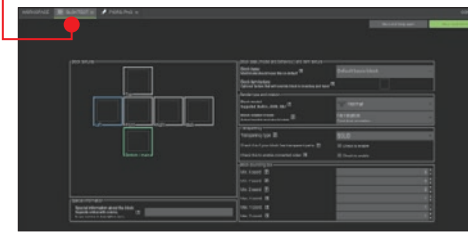
**7** Gdy skończymy edycję i będziemy zadowoleni z efektu, klikamy na **Save this texture** w prawym górnym rogu okna D.



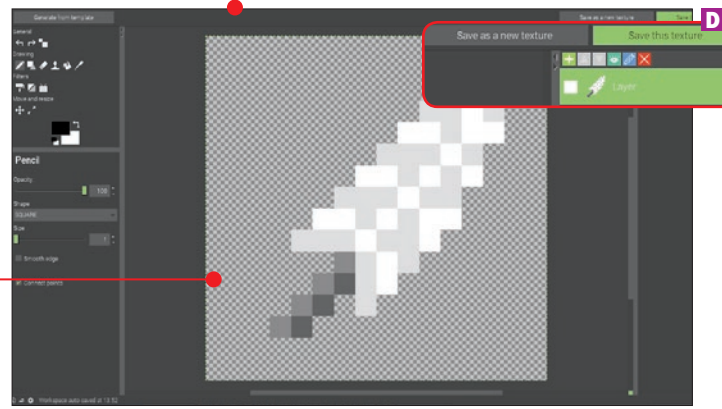

**8** W oknie, które się pojawi, klikamy na **Block**, a następnie w nowym oknie wpisujemy nazwę tekstury.

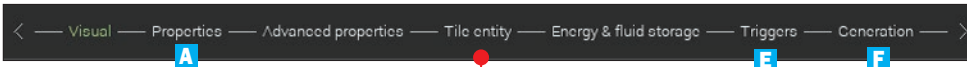


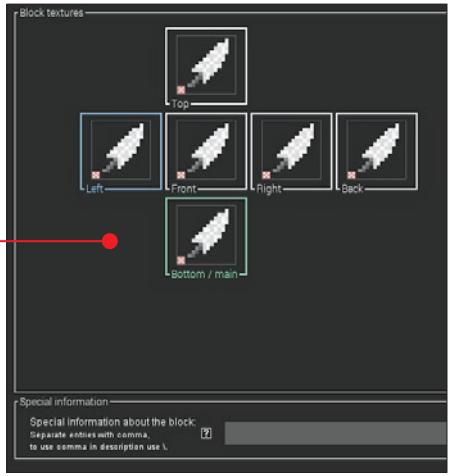

**9** Po stworzeniu tekstury będzie ona widoczna w oknie programu jako zakładka. Dzięki temu w każdej chwili będziemy mogli do niej wracać, by ponownie ją edytować.



**10** Klikając na zakładki, wracamy do planu sześcianu z sześcioma kwadrata-

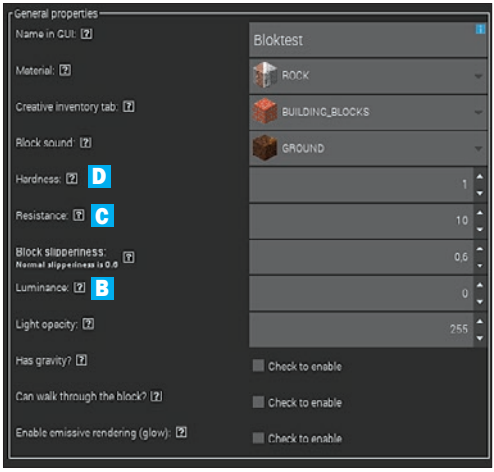





mi (patrz punkt **2**). Gdy teraz klikniemy na jeden z kwadratów, pojawi się już gotowa tekstura pióra. Wybieramy ją – automatycznie wypełni ona wszystkie kwadraty.

**11** Poniżej kwadratów, u dołu okna, znajduje się pasek nawigacyjny naszego bloku. Znajdziemy w nim różne sekcje. Na uwagę zasługuje zakładka **Properties** **A**, gdzie możemy ustawić takie opcje, jak **Luminance** **B** (generowanie światła), **Resistance** **C** (odporność na eksplozje) oraz **Hardness**



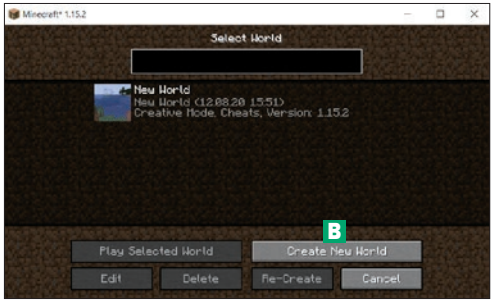

**D** (trudność wydobycia). Można ustawić tu także grawitację i solidność bloku.
Przydatna jest też zakładka **Triggers** **E** – w tym miejscu można zaprogramować sytuacje (na przykład, co się stanie po wydobyciu przedmiotu).
Zakładka **Generation** **F** pozwala ustawić generowanie naszego bloku – na przykład, na jakich wysokościach świata będzie się pojawiać, w jakich ilościach itp. Można też wprowadzić restrykcje dotyczące biomów, na przykład by bloki piór generowały się tylko w biomie pustynnym.

**12** Po dobraniu wszystkich ustawień klikamy w prawym górnym rogu ekranu na **Save mod element**, by zapisać nasz blok. Jeżeli wszystko zostało wykonane poprawnie, będzie można zaimplementować go do naszego świata.

**Nasz blok jest gotowy do zaimplementowania. Jedyne, co pozostało do zrobienia, to kliknąć na zieloną strzałkę w prawo na pasku konfiguracji projektu i startu (patrz strona 45).**

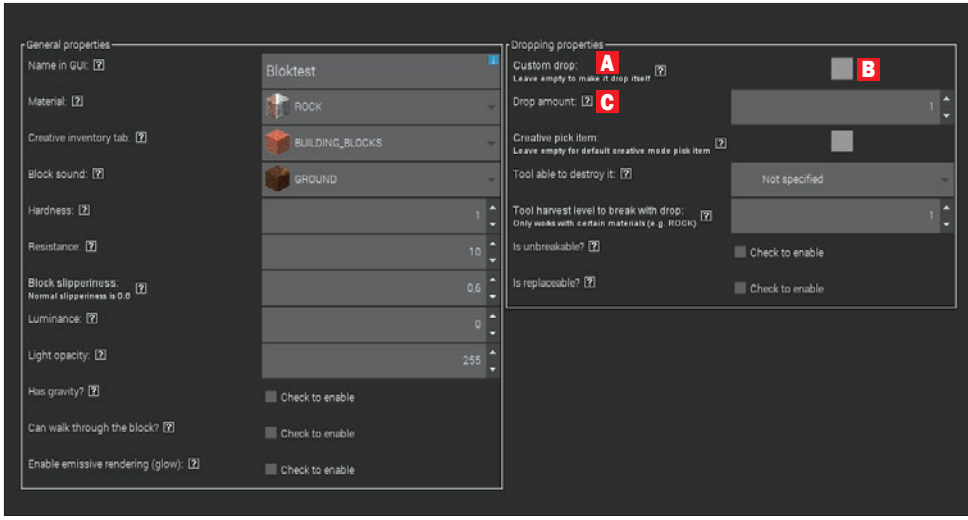
**13** Po uruchomieniu pojawi się okno przygotowania. W zależności od szybkości komputera może się wyświetlać od

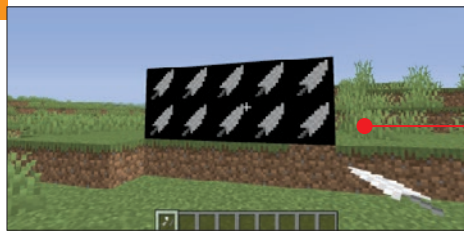
kilkudziesięciu sekund do kilku minut. Po zakończeniu ładowania pojawi się dobrze znane menu startowe Minecrafta.




**14** Naszym zadaniem jest teraz dotrzeć do świata w trybie Kreatywnym – klikamy na **Singleplayer** A, potem **Create New World** B, a następnie **Game Mode: Creative** C i **Create New World** D.

**15** Gdy dotrzemy do utworzonego świata, nasz blok znajdziemy wśród innych bloków na liście. Położenie bloku może być oczywiście edytowane w konfiguracji moda. Jak widzimy, nasz blok znajduje się na końcu listy **Building blocks**.





**16** Po wybraniu go możemy go ustawić w dowolnym miejscu świata.

**17** Pamiętajmy o tym, że chcieliśmy, aby blok piór po zniszczeniu pozostawiał dziewięć piór. Aby to osiągnąć, musimy zatrzymać modyfikację i wrócić do jej edycji. Robimy to, klikając na czerwony kwadrat na pasku konfiguracji projektu i startu. Następnie klikamy na zakładkę naszego moda, przechodzimy do karty **Properties** i otwieramy okno znane z punktu **11**.

**18** W opcji **Custom drop** **A** (ilustracja na poprzedniej stronie) po kliknięciu na kwadrat **B** wybieramy przedmiot, jaki będzie pojawiał się podczas niszczenia bloku – nas interesują pióra. Natomiast poniżej (**Drop amount** **C**) wpisujemy liczbę określającą, ile piór wypadnie z bloku po jego zniszczeniu.

**19** Po zapisaniu wprowadzonych zmian uruchamiamy ponownie moda i sprawdzamy, czy udało nam się sprawić, by po zniszczeniu blok pozostawiał po sobie dziewięć piór. Należy pamiętać, że potrzebujemy włączyć grę w trybie Kreatywnym, jednak później, aby wydobyć przedmiot, należy przełączyć się na tryb Survival (Przetrwanie). Można wykorzystać do tego jedną z komend, którą wprowadzamy za pomocą klawisza /. Wtedy, też w trybie Kreatywnym, dodajemy sobie blok do posiadanego przedmiotu (ekwipunku), umieszczamy go na ziemi, a następnie, korzystając z komendy zmiany trybu gry, wybieramy tryb Survival.

**Pamiętaj, aby po każdej zmianie zapisywać swój projekt, klikając na zielony przycisk w prawym górnym rogu okna programu.**

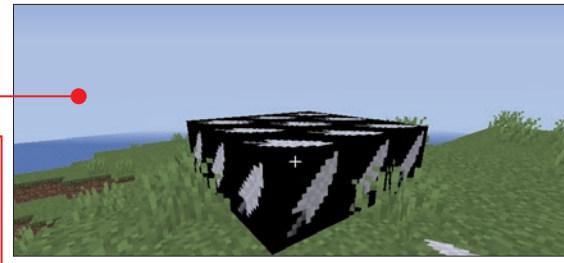
## KOMENDY DO ZMIANY TRYBU GRY:

- **gamemode adventure** – ustawia tryb Przygody
- **gamemode spectator** – ustawia tryb Widza
- **gamemode survival** – ustawia tryb Przetrwanie
- **gamemode creative** – ustawia tryb Kreatywny

Lista przedmiotów do wyboru. Naszym zadaniem jest znalezienie piór

**20** Po uruchomieniu moda i stworzeniu świata w trybie Kreatywnym umieszczamy bloki ●. Następnie przełączamy tryb gry na Survival i niszczymy blok ●.

**Uwaga!** Warto przygotować sobie wcześniej kilof, gdyż zniszczenie bloków ręką nie spowoduje, że pojawią się pióra.



Najwyższa pora przygotować swojego własnego moda. Postaram się pomóc i razem stworzymy mod – Apokalipsę Creeperów. Ja już się boję, a ty?

## BESTIARIUSZ – CREEPER

**Punkty życia:** 20
**Upuszcza:** proch
**Spawn:** wszędzie poza półblokami, szkłem – przy oświetleniu maksimum 8

**Charakterystyka:**
Najbardziej podstawowy mob. Potrafi wybuchać w bliskiej odległości od gracza (odległość 3 bloków). Zaczyna natomiast podążać za graczem, gdy ten zjawi się w odległości 16 bloków.

**Dodatkowe informacje:**
- boi się ocelotów i kotów
- po zabiciu przez szkielet upuszcza płytę
- po trafieniu przez piorun staje się Naładowanym Creeperem o zwiększonej mocy.

## BESTIARIUSZ – WITHER

**Punkty życia:** 300
**Upuszcza:** Netherowa gwiazda
**Spawn:** brak. Należy go stworzyć z Piasku Dusz i Czaszek Witherowego Szkieletu

**Charakterystyka:**
Jeden z najtrudniejszych bossów (trudniejszy od Smoka Endu). Jest wrogo nastawiony do większości stworzeń.

**Dodatkowe informacje:**
walka z nim odbywa się w dwóch etapach – najpierw należy strzelać do niego z broni dystansowych, a potem trzeba walczyć przy użyciu miecza.

# Projekt Apokalipsa Creeperów

## Skąd wziąć model 3D Creepera

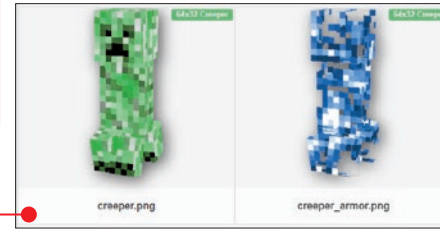
Naszą pracę musimy zacząć od przygotowania tekstury i modelu Creepera. Tworzenie modelu 3D od zera mogłoby być bardzo pracochłonne na początkowych etapach prac z MCreatorem, dlatego warto skorzystać z gotowych projektów mobów. Można je znaleźć na wielu różnych stronach internetowych. W tych wskazówkach wykorzystamy gotowe modele zdeponowane na stronie internetowej **Nova Skin** pod adresem **minecraft.novaskin.me**. Jest to obszerna baza zawierająca przede wszystkim oryginalne modele pochodzące ze świata Minecrafta. Na Nova Skin znajdziemy również olbrzymią społeczność użytkowników udostępniających projekty i pomysły.

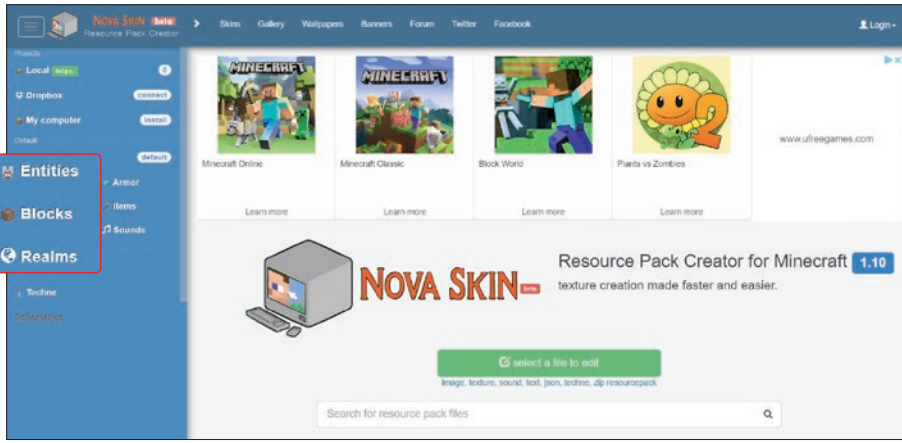
**1** Po wejściu na stronę **minecraft.novaskin.me** zobaczymy jej ekran startowy. Po lewej stronie znajdziemy kategorię **Entities**, w której znajdują się oryginalne modele mobów ze świata Minecrafta.

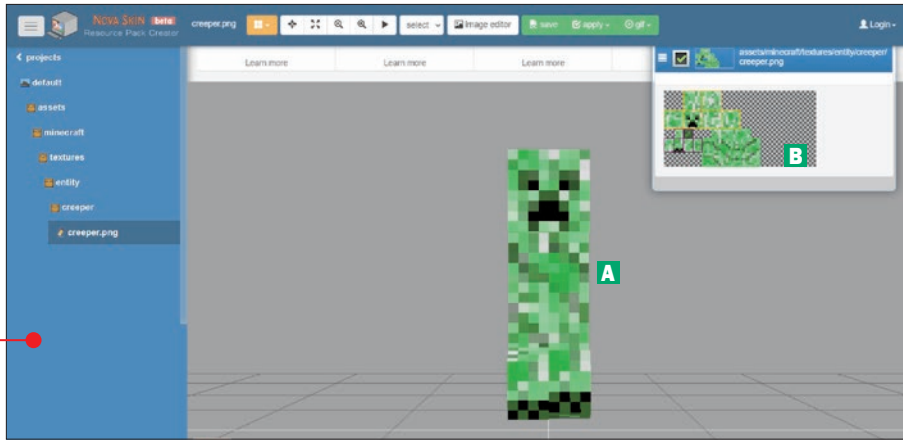
**2** Po kliknięciu na **Entities** pojawią się „skrzynki" mobów. Szukamy Creepera na liście i klikamy na jego skrzynkę. Gdy skrzynka się otworzy, pojawią się dwa dostępne modele Creepera. Wybieramy pierwszy z nich.

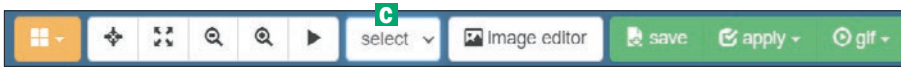
## Z RAPORTU STEVE'A

Naukę projektowania modeli 3D do świata Minecrafta możemy zacząć, korzystając z edytora online **Blockbench**, który pozwala za darmo tworzyć elementy 3D do gier.



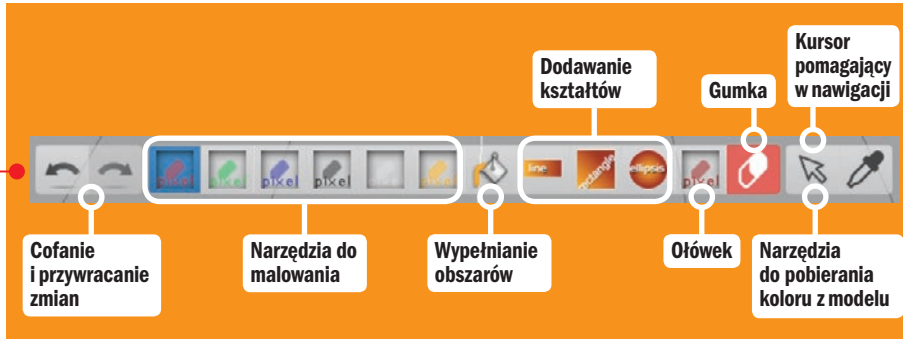



**3** Po wybraniu zielonego Creepera przechodzimy do edytora modelu. W edytorze zobaczymy dwie bardzo ważne rzeczy: model 3D Creepera **A** oraz charakterystyczną siatkę w prawym górnym rogu **B**. Jest to tekstura naszej postaci. Po przewinięciu widoku w dół zobaczymy jeszcze pasek, na którym znajdują się narzędzia do malowania i rysowania. Model obracamy za pomocą lewego przycisku myszy, a przesuwamy prawym. Naszym zadaniem jest teraz skorzystanie z opcji rysowania i malowania, by zmienić model tak, by nam odpowiadał (w dalszej części książki znajdziemy przykładowo zmieniony model, jednak nie musimy się sugerować jego wyglądem). Przydatną metodą edycji modelu jest skupienie się na poszczególnych jego elementach, czyli: najpierw edycja głowy, później edycja tułowia itd.

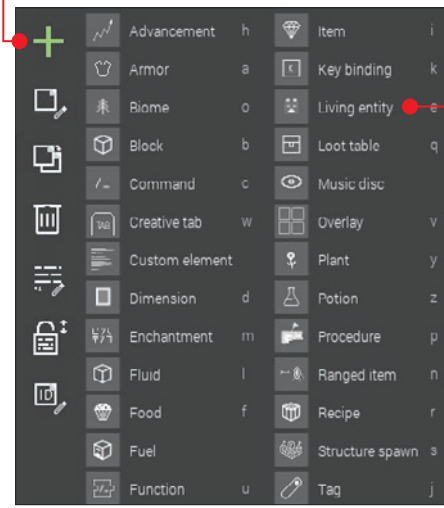
**4** Edytor umożliwia nam wygodną pracę z wykorzystaniem paska menu, na którym znajdziemy poza widokiem pełnego ekranu, zapisywaniem projektu i przybliżaniem i oddalaniem opcję **Select** **C**. Po kliknięciu na **Select** pojawi się rozwijana lista elementów modelu. Po wybraniu jednego z nich model zniknie i zostanie tylko ten, który chcemy edytować **D** (patrz kolejna strona).



**7** Klikamy prawym przyciskiem myszy na grafikę i wybieramy **Zapisz grafikę jako**. Następnie nadajemy plikowi nazwę i wskazujemy lokalizację zapisu na naszym dysku. Po wykonaniu tych kroków możemy już przejść do MCreatora i zacząć tworzyć moda.

**8** Rozpoczynamy od kliknięcia na zielony plus w programie i wybrania z listy opcji **Living entity**, która

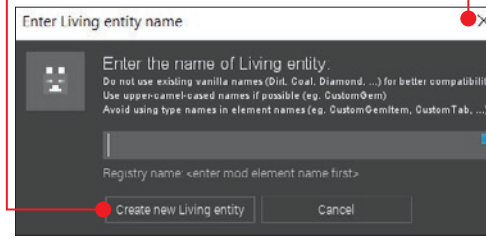


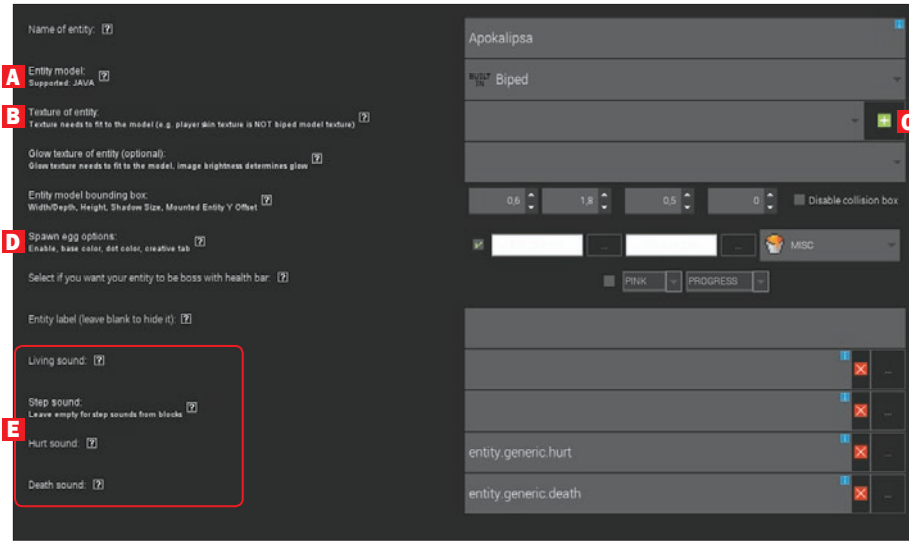

znajduje się w drugiej kolumnie dostępnych opcji.

**5** Oto nasz przykładowy zmieniony model Creepera. Edytując go, zmieniliśmy jego teksturę, ją również znajdziemy w edytorze. Oczywiście model można edytować także przez bezpośrednią edycję tekstury.

**6** Po przygotowaniu modelu klikamy na niebieski przycisk **Save**. W nowym oknie wybieramy sekcję **Download**.

**9** Najpierw zostajemy poproszeni o podanie nazwy naszego moba. Wprowadzamy ją (pamiętając, aby unikać symboli, polskich znaków i spacji). Potem klikamy na **Create new Living entity** i definiujemy właściwości moba.






## Etap 1. Wygląd i dźwięk

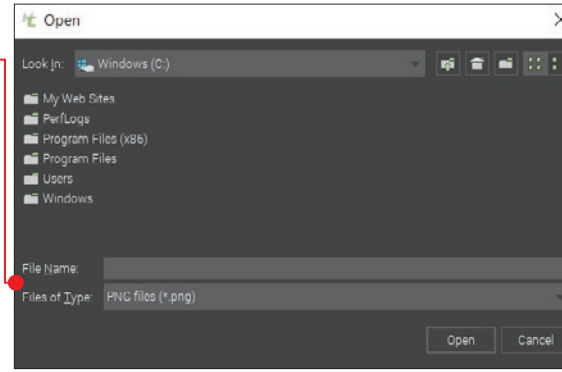
**A Entity model** – w tym miejscu wskazujemy model naszej postaci. Ponieważ chcemy dodać Creepera, to właśnie jego wybieramy z listy. Dzięki temu nasz model w grze będzie mieć właściwy kształt.

**B Texture of entity** – tutaj dodajemy naszą wcześniej pobraną teksturę. Klikamy na mały zielony plus **C** i wgrywamy naszą teksturę. Należy pamiętać o tym, aby od razu potem ją wybrać, gdyż po wgraniu nie wybiera się ona automatycznie.

Kolejną ważną sekcją ustawień jest **Spawn egg option D**, gdzie możemy edytować wygląd jajka spawnującego, które występuje w trybie Kreatywnym.

Ostatnim elementem wartym uwagi w pierwszej części są wydawane dźwięki **E**. Tutaj możemy wprowadzić dźwięk, jaki chcemy, aby wydawało nasze stworzenie.

Są tu do wyboru zarówno dźwięki chodzenia, jak i dźwięki uszkodzenia i śmierci. Wybór dźwięku następuje poprzez kliknięcie na ikonę trzech kropek.



## Etap 2. Zachowanie

Drugi etap to ustawienia związane z zachowywaniem się moba w świecie. Tutaj możemy ustalić, jak na naszego moba będzie wpływać światło, jak szybko będzie się poruszać, czy będzie agresywny, jak zachowywać będzie się w wodzie czy w powietrzu – wszelkie te elementy są oczywiście opcjonalne. Warto jednak poświęcić im uwagę, aby uzyskać jak najlepszy efekt moda.

### Z RAPORTU STEVE'A

**Istnieją stworzenia, które nie mają swojego jaja spawnującego (przyzywającego). Należą do nich Wither, śnieżny golem oraz żelazny golem, ponieważ są one tworzone przez gracza (poprzez układanie określonych bloków w wyznaczonej konfiguracji).**

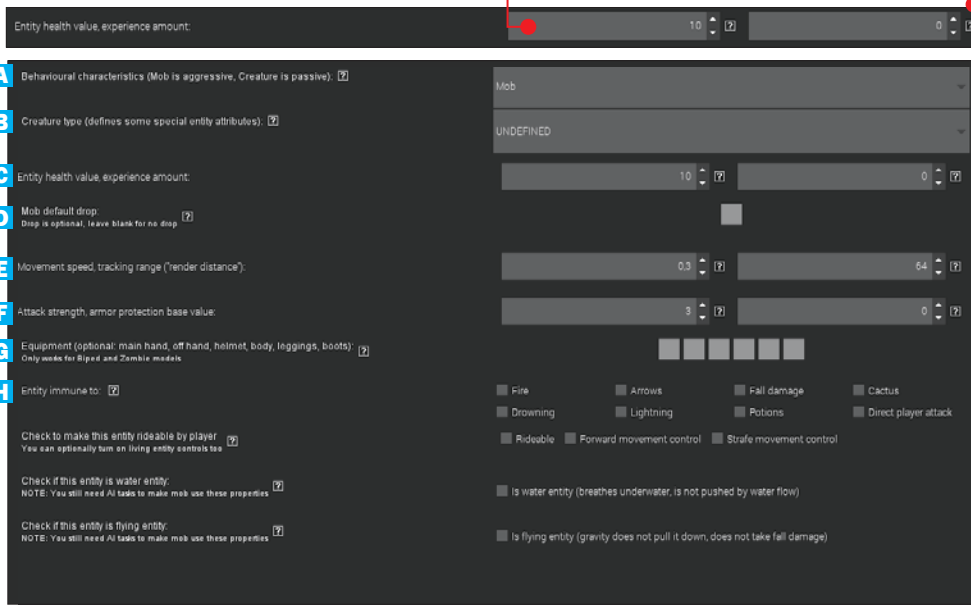
**A Behavioural characteristics** – tutaj do wyboru mamy dwie opcje: **Mob** lub **Creature**. W związku z tym, że nasza postać jest agresywna, wybieramy opcję **Mob**.

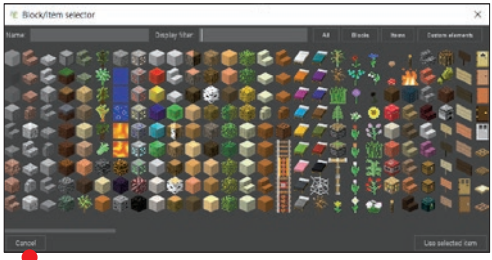
**B Creature type** – w tym miejscu możemy wybrać typ naszej postaci. Każdy typ warunkuje pewne specjalne właściwości. Na przykład wybranie **Arthropod** (stawonóg) sprawi, że nasz Creeper będzie otrzymywał dodatkowe obrażenia od broni zaklętej zaklęciem Zmora stawonogów. W naszym przypadku pozostawiamy tę opcję jako **Undefined** – nasz Creeper nie będzie miał tego typu właściwości.

**C Entity health value, experience amount** – ustawienie punktów życia moba oraz ilości doświadczenia, jakie zdobędziemy po pokonaniu moba. Pierwsze pole ustawia punkty życia, drugie – doświadczenie.

**D Mob default drop** – w tym miejscu decydujemy, co wypadnie z moba po jego po-

W MCreatorze Mob to każda złowroga kreatura (jak Creeper, Zombie czy Utopiec), natomiast Creature to mob pasywny (czyli Świnia, Kurczak czy Kałamarnica)






konaniu. Podobnie jak w przypadku bloku piór decydowaliśmy, co wypadnie po jego zniszczeniu, tak w tym miejscu również deklarujemy, co wypadnie po zabiciu naszego Creepera.

Z listy ● możemy wybrać dowolne przedmioty, które chcemy pozyskiwać po pokonaniu Creepera. Nie wszystkie jednak będą się pojawiać ze względu na brak „dropowego" modelu (jest to model przedmiotu wypadającego podczas zniszczenia jakiegoś elementu – bloku – lub po zabiciu moba). Dlatego też warto sprawdzić różne elementy i wybrać ten, który najbardziej nam się podoba i którego drop jest możliwy.

**E Movement speed, tracking range** – pierwsza opcja odpowiedzialna jest za szybkość poruszania się Creepera, natomiast druga za to, z jakiej odległości będzie on widziany (odległość podana jest w blokach). W pierwsze pole wpisujemy, jak szybko ma się poruszać postać Creepera, w drugim – z jakiej odległości będzie widziana.

W przypadku szybkości i widoczności również warto przetestować różne opcje i wybrać taką, która najbardziej nam odpowiada.

Przykład modeli przedmiotów, które wypadają z mobów lub bloków. Są to najczęściej zminiaturyzowane elementy standardowych odpowiedników. W tej formie nie spotkamy na przykład bloku wody czy lawy, dlatego też nie mogą one stanowić dropu

## Z RAPORTU STEVE'A

Jednym z najszybszych mobów występujących w grze jest Dziecko Zombie. Porusza się dużo szybciej od swojego dorosłego odpowiednika.

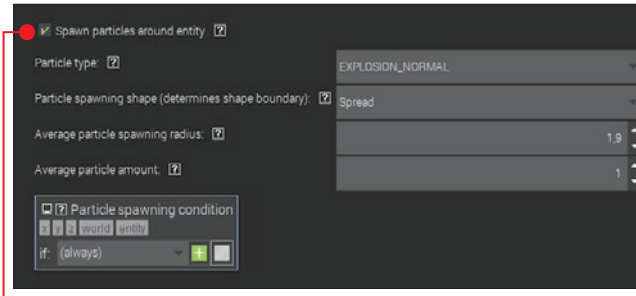
**F Attack strenght, armor protection base value** – tutaj ustawiamy siłę ataku Creepera oraz jego pancerz. Sposób ataku Creepera to oczywiście wybuch przy graczu, więc w tym miejscu określamy siłę jego wybuchu.

**G Equipment** – opcja przedstawiona w formie sześciu kwadratów ●. Ma zastosowanie tylko w przypadku postaci potrafiących nosić przedmioty lub trzymać je w dłoni (w przypadku Creepera nie ma to zastosowania). Każdy z kwadratów dotyczy oddzielnej części ciała.

**1** – Prawa ręka
**2** – Lewa ręka
**3** – Głowa
**4** – Tułów
**5** – Nogi
**6** – Stopy

**H Entity immune to** – tutaj ● wybieramy odporności Cree-

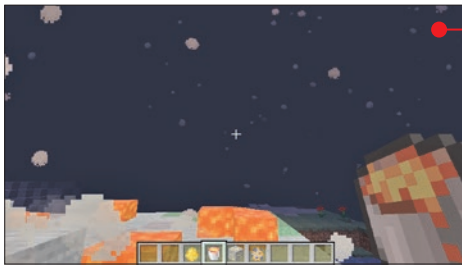
pera na konkretne czynniki: ogień, utonięcie, strzały, błyskawice, obrażenia od upadku, mikstury, kaktusy oraz bezpośrednie obrażenia od gracza. Ważne, aby opcja bezpośrednich obrażeń od gracza (**Direct player attack**) nie była zaznaczona, gdyż mogłoby to spowodować nieśmiertelność Creepera.




## Etap 3. Cząstki

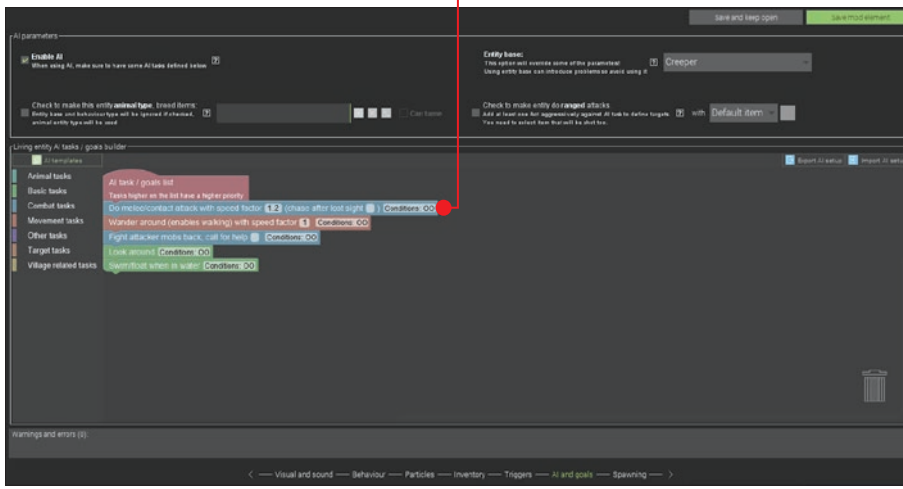
Jest to jeden z najmniej istotnych etapów. Służy jedynie dodaniu cząstek w trakcie życia Creepera oraz w momencie jego przegranej. Można edytować cząstki pod względem graficznym lub skorzystać z gotowych, oryginalnych cząstek pochodzących z plików gry.

Creeper ma cząstki wybuchu. Pojawiają się one dopiero po śmierci Creepe3ra. Zaznaczenie pierwszej opcji ● sprawi, że cząstki te będą widoczne również w trakcie życia Creepera. Aby nadać wyjątkowy wizualnie charakter światu, warto wykorzystać cząstki **campfire_cosy_smoke**. Dzięki temu podczas spawnu Creeperów cały świat wypełni ciemność i charakterystyczne dymy ●.



## Etap 4. AI i cele

Etapy pomiędzy cząstkami a AI i celami należą do zaawansowanych elementów i nie będziemy ich zmieniać w naszym modzie. Można jednak spróbować własnych sił i zaprogramować na przykład inventory postaci. W AI zajmujemy się przede wszystkim sztuczną inteligencją postaci. Za pomocą specjalnych puzzli ● możemy decydować, jak nasz Creeper



## Z RAPORTU STEVE'A

**Sztuczna inteligencja to nic innego jak umysł maszyny/robota/programu. Dzięki różnym wynalazkom można sprawić, że komputer zacznie się uczyć. Przykładem mogą być komputery szachowe, które potrafią pokonać najlepszych graczy.**

będzie się zachowywać w swoim świecie – czy będzie atakować wioski, czy może będzie latać itp. W związku z tym, że nasza postać jest Cree perem, to żeby sobie ułatwić działania, wystarczy w opcji **Entity base** wybrać Creepera. Wtedy też AI naszego Creepera będzie takie samo jak oryginalnej wersji.

### Etap 5. Spawn

W ostatnim etapie decydujemy o tym, jak często, gdzie i w jakich grupach pojawiać się będą nasze Creepery.

Pierwsza **A** i druga **B** opcja (patrz kolejna strona) powinna być zaznaczona – pozwoli to naszej postaci spawnować się. Brak pierwszej opcji sprawiłby, że trzeba by było zbudować naszą postać z jakichś bloków, a zaprojektowanie takiej mechaniki mogłoby być bardzo czasochłonne.

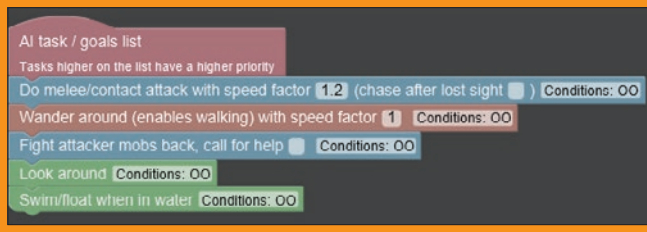
Druga opcja z kolei odpowiedzialna jest za despawn postaci, gdy ta zniknie z pola renderowania (pola, na którym odświeża się świat). Opcja ta powinna być włączona w przypadku każdego moba – jej brak spowodowałby, że każdy pojawiający się mob nie zniknąłby po wyjściu z zasięgu wzroku, a tym samym spowolniłby mocno działanie gry. Opcję tę jednak należy pozostawić włączoną w przypadku bossów i rzadkich mobów – na przykład Wither nie znika, nawet gdy wyjdziemy z zasięgu jego widoczności.

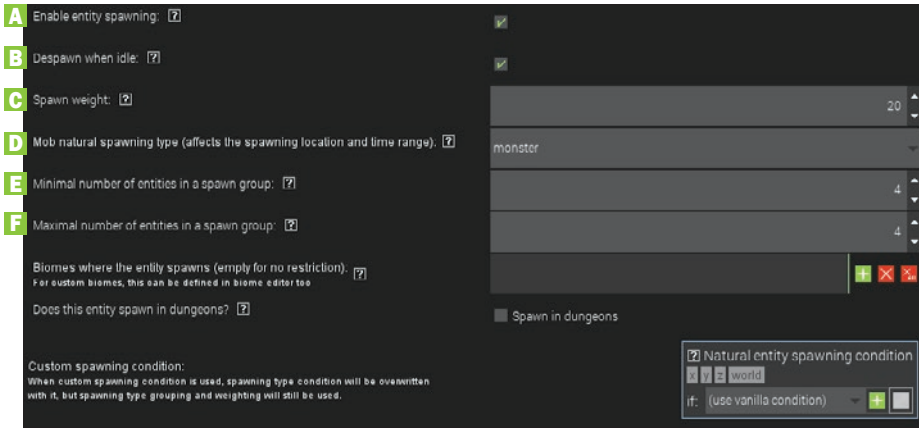
**C** **Spawn weight** – jest to priorytet spawnu. Każdy mob w Minecrafcie ma swoją wartość spawn weightu. Im niższa, tym mob rzadziej będzie się pojawiał na mapie. Jeżeli więc chcemy, żeby dany mob pojawiał się częściej niż inne, to trzeba tę wartość podnieść jak najwyżej, na przykład do 999 (wartość taka sprawi, że w zasadzie wszystkie pojawiające się moby agresywne będą Creeperami). Dla nas jest to istotny element, gdyż Apokalipsa Creeperów będzie polegać właśnie na obronie przed hordami Creeperów.

**D** **Mob natural spawning type** – w tym miejscu należy wybrać opcję **Monster** ze względu na to, że nasz mob będzie się pojawiać tylko w nocy.

Jeżeli chcielibyśmy, aby pojawiał się w dzień, należałoby wybrać **Creature**. Istnieje jeszcze

**To, w jaki sposób działa AI Creepera, można łatwo zobaczyć na skrypcie w formie blokowej zamieszczonym w kategorii AI i cele. Widzimy w nim puzzle odpowiedzialne za atakowanie gracza, błądzenie bez celu, gdy gracza nie ma w pobliżu, czy też pływanie w wodzie. Bloki wykonują się zgodnie z ułożeniem – blok najwyżej położony ma najwyższy priorytet.**

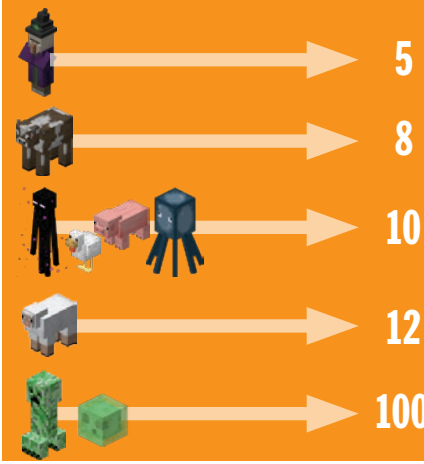





opcja **Ambient** (nietoperze, żaby) oraz **Water Creature** (stworzenia wodne).
**Minimal** E i **maximal** F **number of entities in a spawn group** – obie wartości pozwalają określić zakres ilości zespawnowanych Creeperów.
Zbyt duża liczba mogłaby spowodować spowolnienie komputera, więc dla bezpieczeństwa warto minimalny zakres ustawić na 1, a maksymalny na 20.

## PRIORYTETY SPAWNÓW

Przykładowe priorytety spawnów dla różnych postaci:




## BESTIARIUSZ – SZLAM

**Punkty życia:** 1/4/16 (w zależności od rozmiaru)
**Upuszcza:** Kula szlamu
**Spawn:** niezależnie od poziomu światła do wysokości 39 bloków oraz w biomie bagiennym poniżej 8 poziomu światła na wysokości 51–69 bloków.

**Charakterystyka:**
Występuje w trzech różnych rozmiarach. W najmniejszej formie nie zadaje graczowi obrażeń, tylko go popycha. Potrafi wchodzić po drabinach. Zawsze szuka najkrótszej drogi do gracza, dzięki czemu łatwo się ich pozbyć, na przykład prowadząc je do lawy.

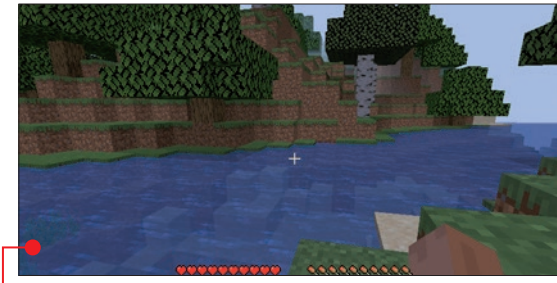
**Dodatkowe informacje:**
- w wyniku błędów szlam może się pojawić w biomach śnieżnych (bardzo rzadko)
- duży osobnik po zabiciu rozpada się na 2–4 mniejsze osobniki
- na mapach superpłaskich jest najczęściej występującym mobem

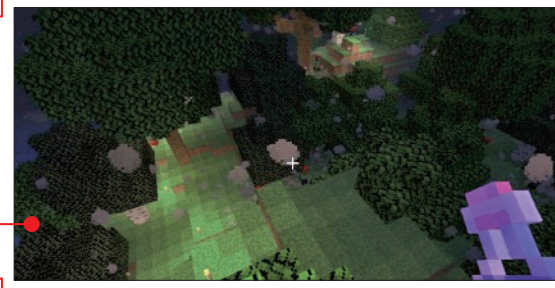
## Testujemy Apokalipsę

Uruchamiamy projekt za pomocą zielonej strzałki i czekamy, aż zostaniemy przeniesieni do świata Minecrafta. Ważne, aby postępować zgodnie z procedurą, która została przedstawiona we wcześniej omówionym projekcie.

Początkowo nie zauważymy żadnej różnicy ●. Nasze Creepery spawnują się tylko w nocy lub w zacienionych miejscach. Dzień należy wykorzystać na pozyskiwanie surowców, budowę schronienia i przygotowanie żywności. Podczas zwiedzania świata można się już jednak natknąć na pojedyncze Creepery ●.

Po zmierzchu można się natomiast spodziewać hord Creeperów ●. Ich obecność będzie zaznaczona wszechobecnymi dymkami ●. Ilustracje obok pokazują przebieg naszej Apokalipsy.



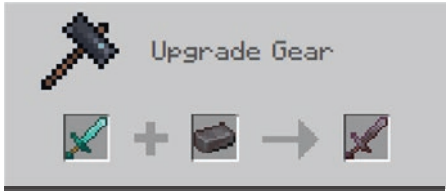


**Podczas testowania zwróć uwagę, czy kompilacja projektu zachodzi poprawnie. W przypadku błędu gra nie powinna się otworzyć. Zawsze jednak warto śledzić wszystkie procesy pojawiające się podczas kompilowania. W prawym dolnym rogu programu znajduje się informacja o tym, jakie zadanie w tym momencie jest wykonywane. Dodatkowo w przypadku błędów pojawi się specjalny dziennik informujący o tym, gdzie pojawił się błąd.**

Task :runClient

# Projekt Miecz Niebios

Podczas naszej przygody udało nam się już edytować bloki i nadawać im pewne właściwości. Dodatkowo pracowaliśmy na istniejących mobach, edytując ich zachowania, wygląd i zaawansowane aspekty działania. Kolejnym tematem, na którym się skupimy, będzie dodanie i edycja miecza. Po-




**Ulepszanie miecza diamentowego Netherytem prowadzi do stworzenia Netherowego Miecza**

staramy się przygotować elementy graficzne (dobrze znane nam tekstury), mechanikę działania i właściwości. Poznamy także nieznaną nam dotąd funkcję MCreatora, dzięki której można programować obiekty.

Miecz to podstawowa broń podczas walk z agresywnymi mobami. W zależności od materiałów, z jakich został zbudowany, ma różną wytrzymałość, zadaje inną liczbę obrażeń itp. Dotychczas najsilniejszym mieczem był miecz diamentowy, najsłabszym zaś miecz drewniany. W wersji 1.16 gry dodano jednak miecz netherythowy, który powstaje w wyniku ulepszenia miecza diamentowe-

## Z RAPORTU STEVE'A

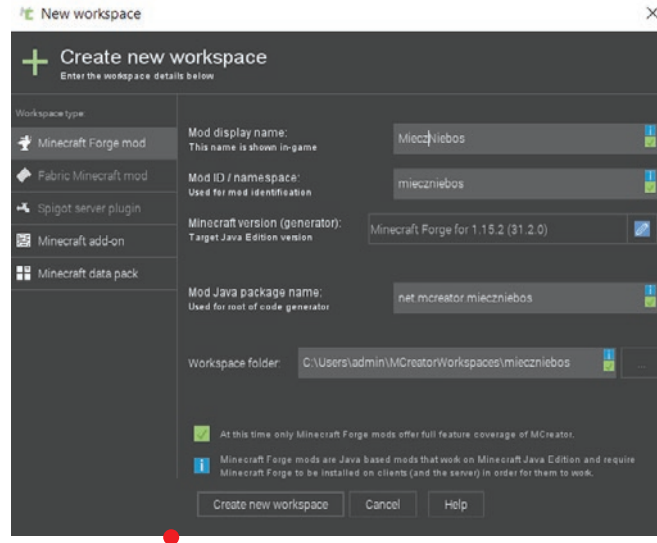
Netheryt to bardzo rzadki materiał występujący w Netherze. Jest bardzo wytrzymały i służy głównie do ulepszania diamentowych przedmiotów. To jedyny materiał w grze odporny na działanie lawy. Przedmioty z niego stworzone nie ulegają zniszczeniu po wpadnięciu do lawy.

## CHARAKTERYSTYKA MIECZY

| MATERIAŁ | WYTRZYMAŁOŚĆ | OBRAŻENIA |
|---|---|---|
| drewno | 60 | |
| złoto | 33 | |
| kamień | 132 | |
| żelazo | 251 | |
| diament | 1562 | |
| Netheryt | 2031 | |

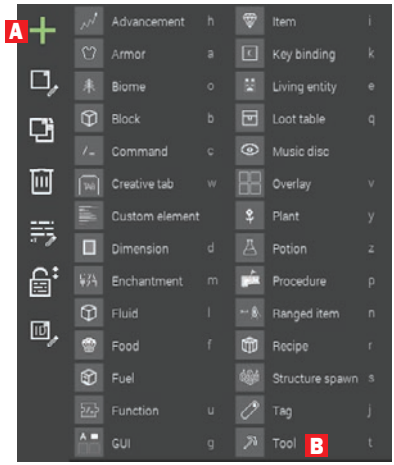
go. Charakteryzuje się on znakomitą wytrzymałością i zadawaniem wyjątkowo dotkliwych obrażeń. Wytrzymałość mieczy stworzonych z różnych materiałów, a także inne właściwości przedstawiono w tabeli. Warto pamiętać, że Netheryt znajduje się w najnowszej aktualizacji, przez co może być niedostępny dla graczy, którzy korzystają ze starszych wersji Minecrafta.

W naszym przypadku samodzielnie będziemy decydować o tym, w jakim przedziale będzie mieścić się wytrzymałość miecza, liczba zadawanych obrażeń, szybkość itp.



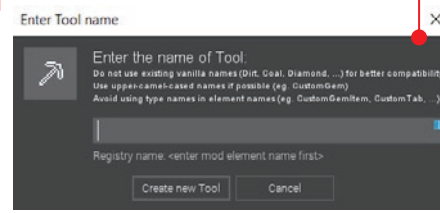

Projekt Miecz Niebios będzie polegać na dodaniu nowego miecza do gry, który podczas atakowania przeciwników będzie przywoływać błyskawice, a każdy uderzony nim mob będzie unosić się w górę z charakterystycznym świeceniem.

**1** Pracę z projektem zaczynamy od utworzenia nowego miejsca do pracy (**Workspace**) ●.

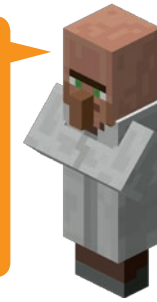



**2** Następnie klikamy na zielony plus **A** i z rozwijalnego menu wybieramy **Tool** **B**. (Zakładka **Tool** służy do dodawania wszelkiego rodzaju narzędzi i broni. Jeżeli więc kiedykolwiek będziemy potrzebować zaprojektować kilof, edytować łopatę czy motykę albo broń, to jest to właściwe miejsce).

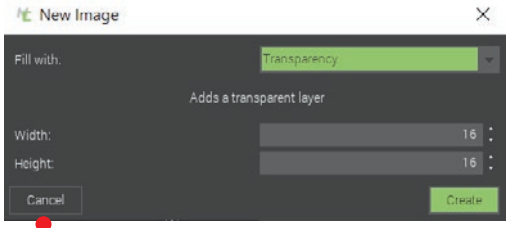
**3** Po wybraniu opcji **Tool** naszym kolejnym krokiem jest nazwanie przedmiotu ●.




**Pamiętaj, żeby nie używać: polskich znaków, spacji i symboli oraz już istniejących nazw z Minecrafta, czyli na przykład DiamondSword czy Dirt. Mogłoby to sprawić, że obecnie istniejące elementy w grze będą działały nieprawidłowo.**



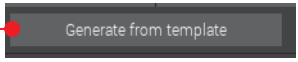
Nazywamy nasz przedmiot, pamiętając o zasadach nadawania nazw plikom i obiektom w programie MCreator, a następnie zostajemy przeniesieni do pierwszej karty właściwości, a mianowicie do karty **Wyglądu**, gdzie będziemy mogli zaprojektować teksturę miecza.




**4** Klikamy na pusty kwadrat z napisem **Tool Texture**. Pojawi się opcja wyboru tekstury naszego miecza. Ponieważ nie mamy żadnej tekstury, będziemy musieli stworzyć własną. Możemy skorzystać z gotowych modeli już istniejących mieczy lub w całości przygotować własny.

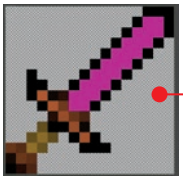
**5** Aby stworzyć teksturę, w nowym oknie wybieramy opcję **Create texture** w lewym, górnym rogu.

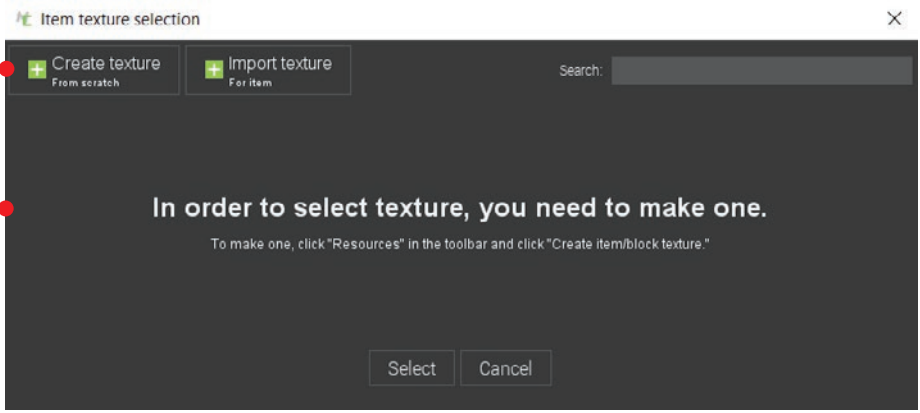
**6** Kolejne okno pozwala nam na wybór wielkości tekstury oraz jej wypełnienia. Najwygodniej w tym przypadku pozostać przy domyślnych ustawieniach. Po kliknięciu na zielony przycisk **Create** zostaniemy przeniesieni do edytora graficznego, w którym można narysować miecz lub skorzystać z gotowej grafiki i jedynie ją edytować.

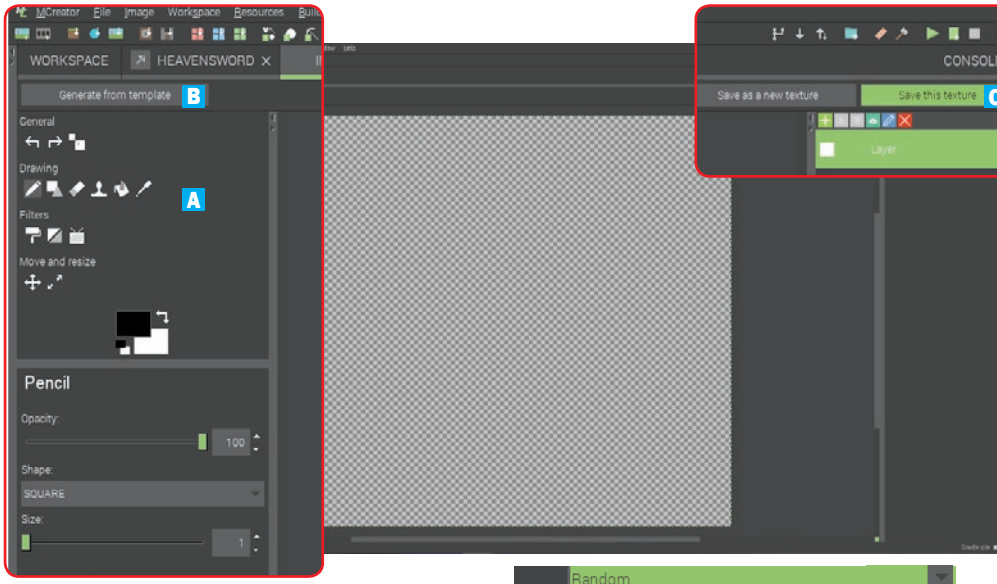



W przypadku tworzenia własnej grafiki od podstaw pracujemy z przybornikiem po lewej **A**. Natomiast aby edytować istniejące grafiki, klikamy na przycisk **B**.

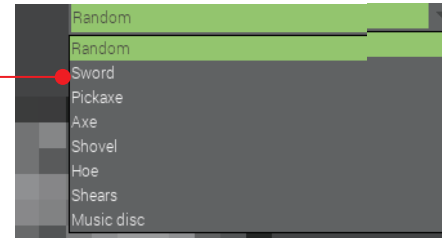
**7** Po kliknięciu na ten przycisk pojawi się okno, w którego prawym górnym rogu znajdziemy



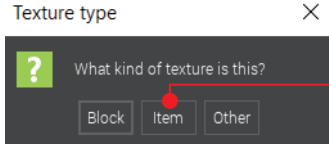




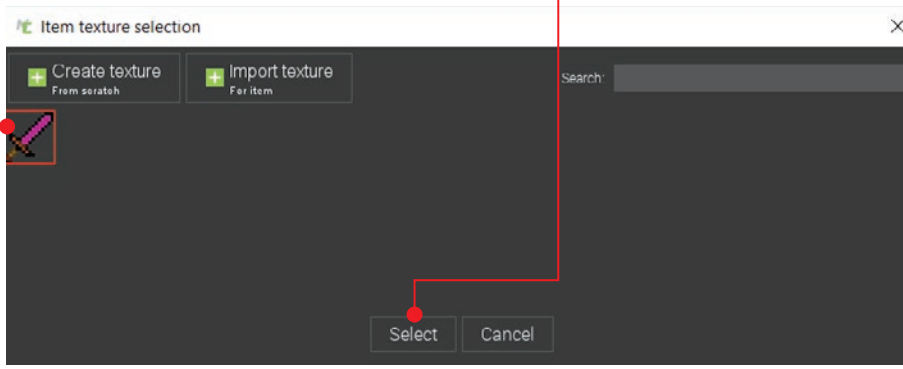

rozwijaną listę dostępnych template'ów. Wybieramy z listy **Sword** i klikamy na **Randomize** i **Create**. Następnie edytujemy miecz, który pojawi się w edytorze.



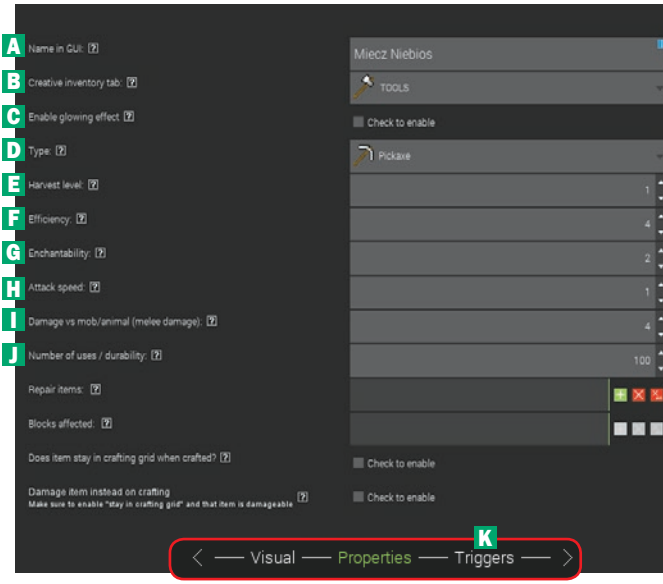

**8** Miecz możemy dowolnie edytować, aż otrzymamy pożądany efekt. Gdy uznamy, że jest gotowy, klikamy na **Save this texture** C w prawym górnym rogu. W kolejnym kroku musimy z nowego okna wybrać, jaki rodzaj tekstury przygotowaliśmy. W naszym przypadku wybieramy **Item**.




**9** Nadajemy nazwę teksturze, pamiętając, by nie używać spacji. Gdy już ją zapiszemy, wracamy do okna z wyborem tekstury – można już ją wybrać. Klikamy na przycisk **Select**.

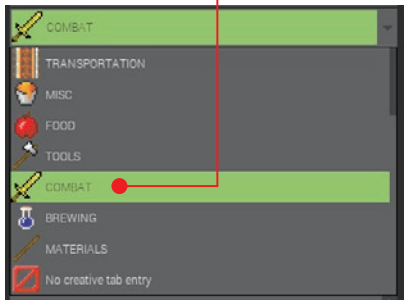


**10** Teraz dobieramy właściwości naszego miecza. Poniżej opis każdej z dostępnych opcji:




**A Name in GUI** – nazwa przedmiotu w całej grze. W każdym menu, podmenu i w innych miejscach miecz pojawiać się będzie właśnie pod tą nazwą.

**B Cretive inventory tab** – miejsce, w którym nasz miecz będzie się pojawiać w ekwipunku trybu Kreatywnego (tutaj wybieramy **Combat** z listy rozwijanej).




**C Enable glowing effect** – w tym miejscu możemy ustawić świecenie obiektu (tak jak w przypadku na przykład zaklętych przedmiotów lub złotego jabłka).

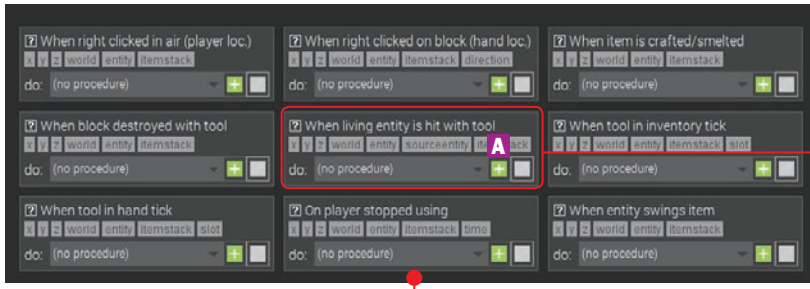
**D Type** – typ obiektu. Wybranie określonego typu sprawi, że niektóre z parametrów poniżej nie będą wykorzystywane, na przykład nie wszystkie obiekty służą do wydobywania przedmiotów. Z rozwijanej listy wybieramy przedostatnią opcję **Special**. Jest to właściwość charakterystyczna dla przedmiotów tworzonych za pomocą MCreatora.

Następne opcje są definiowane przez konkretne wartości liczbowe.

**E Harvest level** – oznacza poziom wydobycia. Określa zakres bloków, jakie mogą być wydobywane za pomocą danego narzędzia. Istnieje skala, która określa, które elementy będą wydobywane danym narzędziem. Na przykład:

**0** – siła wydobycia ręką lub drewnianym narzędziem
**1** – siła wydobycia kamiennym narzędziem
**2** – siła wydobycia żelaznym narzędziem
**3** – siła wydobycia diamentowym narzędziem.

Na tej podstawie można określić siłę wydobycia dla naszego narzędzia. Na przykład wybranie 0 sprawiłoby, że nie moglibyśmy wydobyć naszym narzędziem rudy diamentu. W związku z tym, że nasz miecz nie służy do wydobywania, pozostawiamy tu wartość domyślną.




**F Efficiency** – szybkość wydobycia materiałów. Również w naszym przypadku pozostawiamy tu wartość domyślną (nasz miecz nie służy do wydobycia).

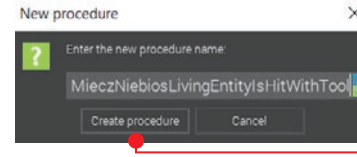
**G Enchantability** – poziom zaklinania przedmiotu. Im wyższa wartość tego parametru, tym mocniejsze zaklęcia będziemy otrzymywać podczas zaklinania. Ustawiamy dowolną wartość, jaką uważamy za odpowiednią.

**H Attack speed** – szybkość ataku, która jednocześnie wpłynie na zadawane obrażenia. Wybieramy wartość, jaka nam odpowiada.

**I Damage vs mob/animal** – zadawane obrażenia. Również tutaj możemy ustawić wybrane przez nas wartości.

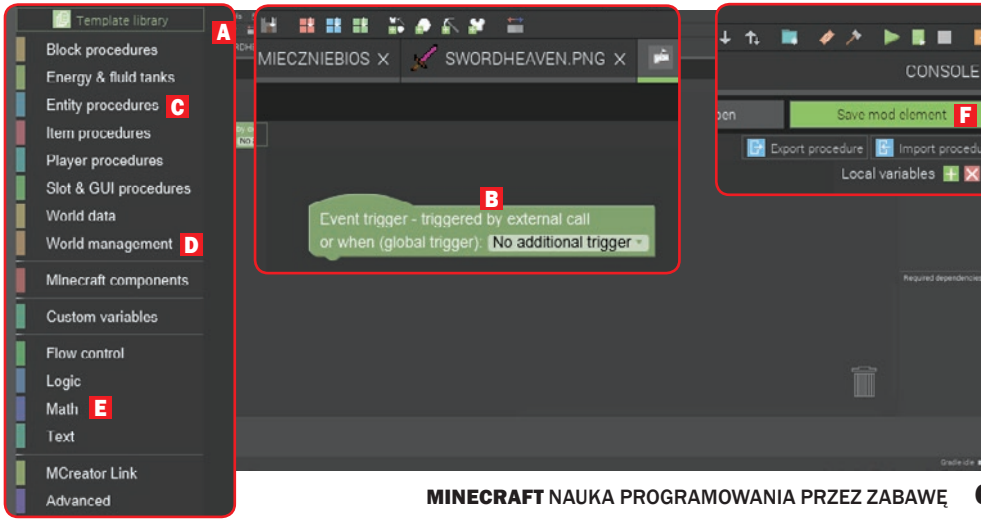
**J Number of uses/durability** – liczba użyć naszego przedmiotu (wytrzymałość).

**11** Po ustawieniu tych podstawowych wartości przechodzimy do kolejnej zakładki – **Triggers** (Wyzwalanie) **K**. Tu możemy edytować elementy związane z właściwościami przedmiotu podczas walki, używania, tworzenia na crafting table itp. Wybieramy pozycję ●, klikając na zielony plus **A**.



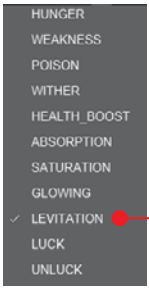

**12** Zostajemy poproszeni o podanie nazwy, zatwierdzamy ją, klikając na przycisk **Create procedure** ●. W ten sposób przechodzimy do kolejnej, bardzo ważnej części programu MCreator, czyli do panelu programowania.

**13** Okno do programowania składa się z dwóch istotnych elementów – kategorii bloków **A** oraz obszaru roboczego z bazowym blokiem **B**. Poznajmy funkcje, które będą potrzebne do stworzenia naszego miecza. Bloczek startowy pozwala na uru-



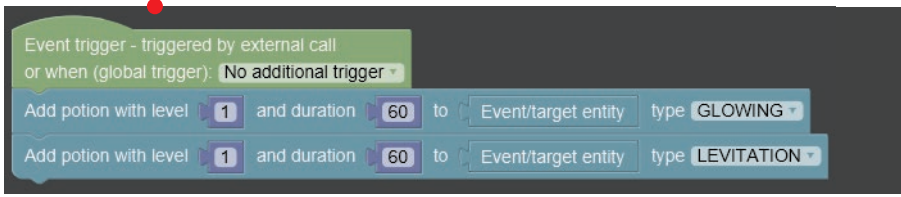
chomienie skryptu, gdy zostanie spełniony jakiś warunek. W naszym przypadku jest to uderzenie mieczem moba.
Kategorie oznaczone na czerwono pozwalają wybrać różne puzzle, które połączone odpowiednio ze sobą stworzą program. Naszym zadaniem będzie dodanie do miecza uderzeń błyskawic oraz zaprogramowanie, aby każdy mob pod wpływem miecza unosił się do góry (lewitacja) w charakterystycznej świetlnej otoczce (glowing).

**14** Zaczynamy od wyboru kategorii **Entity procedures** C (na poprzedniej stronie). Znajdziemy w niej wszelkie bloki związane z programowaniem mobów – spawny, despawny, obrażenia itp. Na liście tych bloków szukamy bloku **Add potion with level...** A. Sprawia on, że mob uderzony przez przedmiot otrzymuje na określony czas konkretny efekt o wybranej sile. Przy napisie **SPEED** B znajdziemy rozwijaną listę. Wybieramy z niej efekt **LEVITATION**. Dzięki temu otrzymamy blok wyglądający tak C.

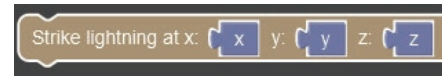



**15** Procedurę powtarzamy, aby uzyskać drugi blok, tym razem z efektem **GLOWING** D.

**16** Bloki łączymy ze sobą, tworząc następujący skrypt.



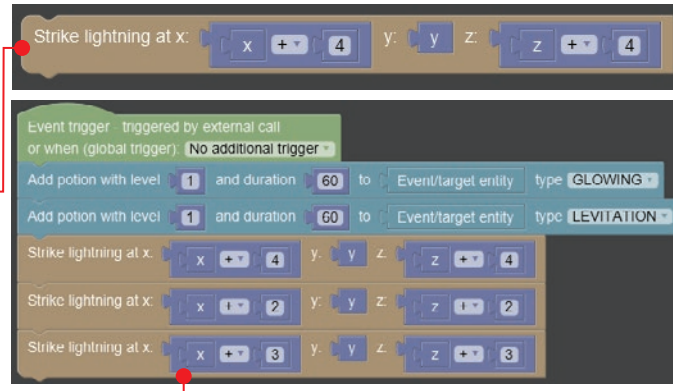

**17** Kolejnym etapem jest dołączenie do naszych bloków elementów związanych z pojawianiem się błyskawic po zaatakowaniu moba. Aby to zrobić, należy zajrzeć do kategorii **World management** D (na poprzedniej stronie), w której znajdziemy elementy związane z pogodą, globalnymi spawnami czy czasem gry. Szukamy w tej kategorii bloku **Strike lightning at...**, który sprawi, że w koordynatach X, Y, Z uderzy błyskawica.




**18** X, Y, Z to koordynaty naszej postaci. Gdybyśmy pozostawili je tak jak teraz, błyskawica uderzałaby w nas. Dlatego też wyciągamy puzzle **X**, **Z** z bloku **Strike lightning at...** i zastępujemy je blokami dodawania z kategorii **Math** E. Jest to kategoria, w której znajdziemy proste obliczenia matematyczne. Bloki dodawania umieszczamy w miejscu koordynat X i Z (Y może pozostać takie, jakie jest, czyli poziom wysokości, na jakim znajduje się nasza postać). W efekcie uzyskamy następujący blok.

**19** W puste miejsca wprowadzamy liczby, które również znajdziemy w kate-

gorii **Math**. Liczby te zostaną dodane do koordynatów. Warto dodawać niewielkie liczby, aby błyskawice pojawiły się w naszej okolicy, na przykład liczbę **4**.



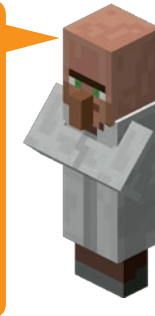

**20** Procedurę powtarzamy kilkakrotnie, aby otrzymać kilka błyskawic. Ważne, aby zmienić liczbę dodawaną do X i Z, gdyż w przeciwnym razie błyskawice pojawiałyby się w tym samym miejscu. Wprowadzone bloki łączymy ze skryptem stworzonym wcześniej – w efekcie otrzymamy gotowy program.

**21** Po stworzeniu programu możemy przejść do jego testowania, klikając na **Save mod element** F. Uruchamiamy program. Tak jak w przypadku poprzednich projektów należy pamiętać, żeby uruchomić grę w trybie **Creative**, aby mieć łatwy dostęp do nowo stworzonego obiektu.

**22** Po uruchomieniu gry w trybie Kreatywnym sprawdzamy, czy nasz projekt działa prawidłowo. Nasz miecz w ekwipunku trybu Kreatywnego odnajdziemy w kategorii **Combat** (Bronie).

**23** Spróbujmy wykorzystać go w walce i zobaczyć, jak się sprawuje przeciwko mobom. Jak widzimy, koń uderzony mieczem uniósł się góry. Wokół niego widzimy również charakterystyczną obwódkę. Płomień jest efektem uderzenia w niego błyskawicą. W ziemię uderzają trzy błyskawice. Należy uważać, grając w trybie Survival, aby samemu nie paść ofiarą uderzenia.

**Uważam, że radzisz sobie świetnie i mam nadzieję, że jeszcze kiedyś mnie odwiedzisz. Ja, niestety, muszę wracać do mojej wioski – pracujemy z innymi Osadnikami nad modem, który raz na zawsze zlikwiduje problem Zombie. Wpadnij jeszcze kiedyś, to może pokażę ci, jak takiego moda zrobić. Powodzenia w dalszej wędrówce!**



# 4 Język Python a modyfikowanie Minecrafta

**Python to bardzo popularny język programowania. W tym rozdziale poznamy komendy Pythona pozwalające na ingerencję w świat Minecrafta. Dowiemy się też, jak wywołać stworzony kod w świecie Minecrafta**

Języki programowania nie służą tylko do tworzenia nowych programów, ale pozwalają także modyfikować już istniejące aplikacje, w tym także gry, nadając im całkowicie nowe funkcje. Naszą przygodę z modyfikowaniem Minecrafta za pomocą poleceń Pythona rozpoczniemy od krótkiego przeglądu najpopularniejszych języków programowania.

**Witaj. Jestem wędrownym handlarzem. Zwykle podróżuję z lamą, jednak zgubiłem ją, pewnie jest gdzieś na końcu rozdziału. Proponuję współpracę: ty pomożesz mi się tam przedostać, a ja w trakcie podróży pokażę ci, jak działają języki programowania i jak modyfikować świat Minecrafta za pomocą języka Python. Przygotuj się do wspólnej podróży!**

## Języki programowania

Język programowania pod pewnymi względami przypomina język naturalny. Jest to zbiór zasad, które opisują sposób porozumiewania się z komputerem. Odpowiednio grupując i rozmieszczając różnorodne elementy, takie jak komendy, funkcje, zmienne itp., tworzymy program komputerowy. Jest to sposób na porozumiewanie się z komputerem i przekazywanie mu informacji, jakie funkcje ma wykonywać. Podobnie jak uczymy się języka naturalnego (hiszpańskiego, francuskiego, włoskiego itp.), aby porozumiewać się z innymi ludźmi, tak uczymy się języka programowania, aby komunikować się z komputerem.

Języki programowania mają dość długą historię. Pierwszym językiem programowania był Fortran – powstał w roku 1957.

Obecnie do najpopularniejszych języków programowania należą Java Script, Python i Java. Z roku na rok kolejne języki programowania zdobywają popularność, jednak od kilku lat ścisła czołówka w zasadzie się nie zmienia. Podobnie jak język angielski, francuski czy hiszpański są najpopularniejszymi językami wybieranymi do nauki, tak samo programiści mają swoje ulubione typy wśród języków programowania (pełny ranking – poniżej). Język programowania, w którym będziemy programować i tworzyć w tym rozdziale, to Python.

**Najpopularniejsze języki programowania w roku 2020 według firmy analitycznej RedMonk**

### NAJPOPULARNIEJSZE JĘZYKI PROGRAMOWANIA W ROKU 2020*

| | | | |
|---|---|---|---|
| 1. JavaScript | 6. C++ | 11. Swift | 15. Shell |
| 2. Python | 7. Ruby | 12. Objective-C | 17. PowerShell |
| 2. Java | 7. CSS | 13. Scala | 18. Perl |
| 4. PHP | 9. TypeScript | 13. R | 19. Kotlin |
| 5. C# | 9. C | 15. Go | 20. Haskell |

* Według firmy analitycznej RedMonk

# Python w Minecrafcie: przygotowania

Python to już prawie trzydziestoletni język programowania, który w ostatnich latach zdobywa olbrzymią popularność. Jest to bardzo czytelny język i dlatego zaleca się go osobom, które dopiero rozpoczynają swoją przygodę z programowaniem.

Wykorzystanie Pythona w programowaniu Minecrafta wymaga przede wszystkim skonfigurowania i zainstalowania kilku elementów, bez których nie będziemy w stanie ingerować w świat Minecrafta. Zaczniemy więc od przygotowania komputera i odpowiednich programów do pracy.

Oto, czego będziemy potrzebowali, żeby język Python prawidłowo funkcjonował w Minecrafcie:

- zainstalowana najnowsza wersja Javy SDK,
- program IDLE,

- API Minecrafta,
- lokalny serwer Minecrafta,
- gra Minecraft.

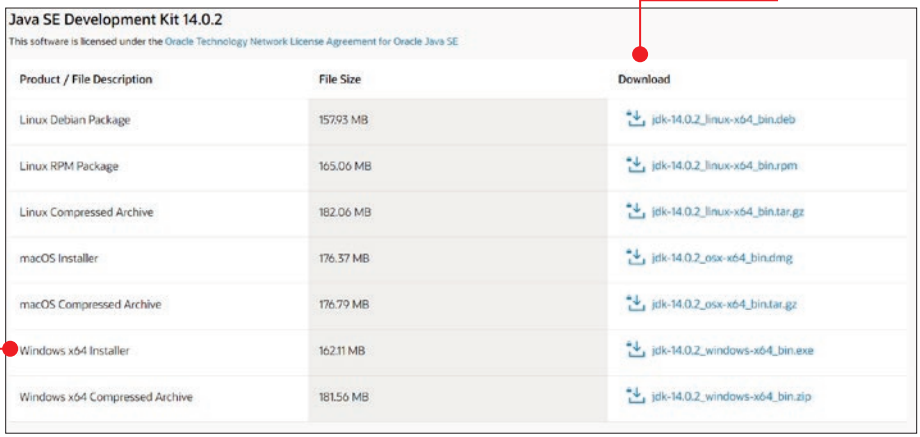
Przyjrzyjmy się dokładniej każdemu z tych elementów.

## Java SDK

Zaczynamy od instalacji najnowszej wersji **Javy SDK** (**W KŚ+**). Jest to krok konieczny do prawidłowego działania lokalnego serwera Minecrafta oraz ingerowania w samą strukturę gry. Javę SDK znajdziemy w KŚ+ (ksplus.pl). Możemy też pobrać ją ze strony: **oracle.com/java/technologies/javase-downloads.html**.

Po wejściu na stronę wybieramy opcję **JDK Download**, która pozwoli nam przejść do wyboru wersji, jaką chcemy pobrać (zależnie od naszego systemu operacyjnego).

Dalsze kroki będą przedstawiane dla opcji **Windows x64 Installer**. Można oczywiście pobrać archiwum i rozpakować je na komputerze – wszystko zależy od tego, co nam bardziej odpowiada. Klikamy na nazwę pliku w kolumnie **Download**.



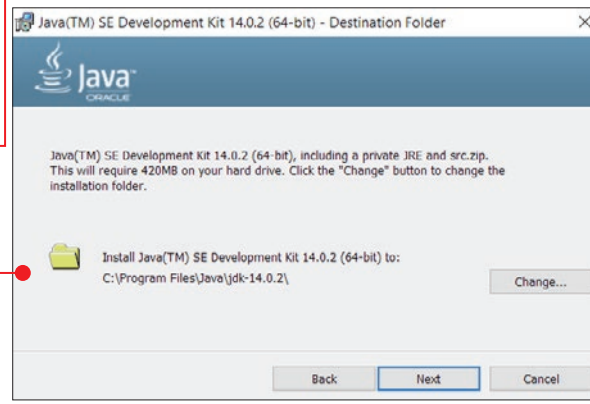
Zostaniemy jeszcze poproszeni w nowym oknie o zaakceptowanie licencji. Po zaakceptowaniu licencji przycisk **Download** zmieni kolor na zielony. Rozpocznie się proces pobierania instalatora. Po ukończeniu pobierania uruchamiamy plik instalacyjny dwukrotnym kliknięciem.
Uruchomimy tym samym proces instalacji.

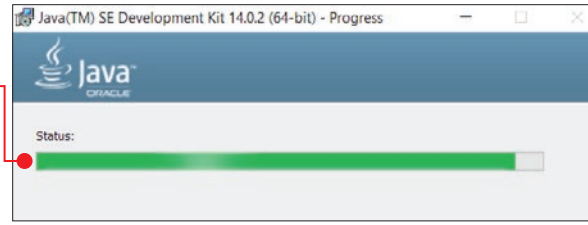
**1** Na pierwszym ekranie instalatora klikamy po prostu na przycisk **Next**.

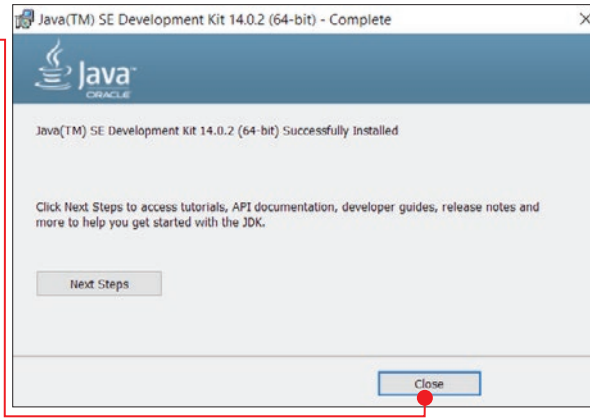
**2** Następnie wybieramy miejsce, w którym zostanie zainstalowany program. Można dokonać wyboru miejsca docelowego lub pozwolić instalatorowi zdecydować za nas i zaakceptować zaproponowaną opcję.

**3** Instalator od razu rozpocznie instalowanie programu. O stanie instalacji będzie informować nas pasek postępu.

**4** Po zakończeniu instalacji zobaczymy przycisk **Next Steps**, który prowadzi do dokumentacji i tutoriali. Zamykamy okno instalatora, klikając na przycisk **Close**.



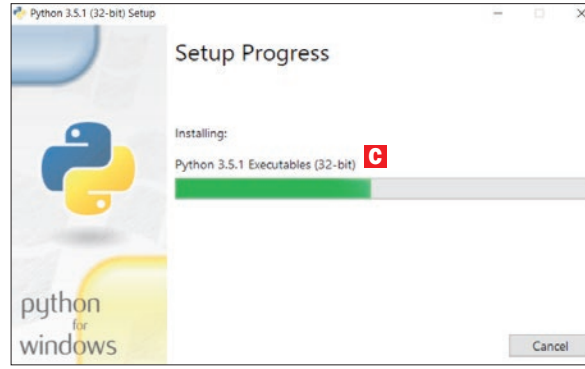



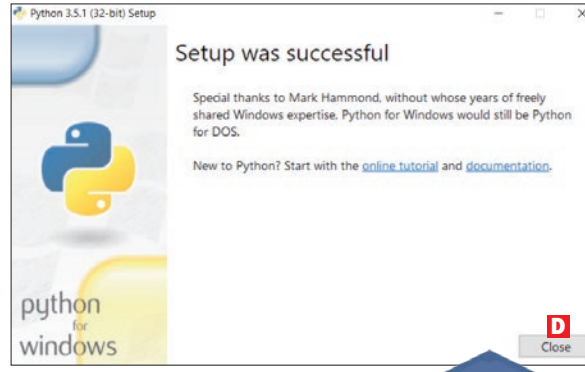




## Program IDLE

IDLE to program służący do pracy z wykorzystaniem języka programowania Python, znajdziemy go na dysku po zainstalowaniu Pythona.
Dzięki IDLE będziemy mogli tworzyć różnego rodzaju programy, które między innymi będziemy mogli zaimplementować do świata Minecrafta.

**Gdy chcemy napisać opowiadanie, to my, handlarze, najczęściej wykorzystujemy do tego pióro, atrament oraz papier. Napisanie programu komputerowego również wymaga pewnych elementów. Stosuje się do tego coś, co nazywamy zintegrowanymi środowiskami programistycznymi, czyli – IDE.**




**1** Rozpoczynamy instalację **Pythona** (DVD-KOD: 010) z płyty dołączonej do książki, na przykład klikając dwukrotnie na plik instalacyjny **Python-3.5.1.exe*** w folderze płyty lub na **Instaluj** w menu płyty przy opisie Pythona.

**2** Na ekranie startowym musimy zacząć od kliknięcia na pole przy **Add Python 3.5 to PATH** A i zaznaczenia go. Dopiero potem możemy przejść dalej, klikając na **Install Now** B.

**3** Zacznie się instalacja Pythona C, która trwa zwykle od kilkudziesięciu sekund do kilku minut. Po jej zakończeniu pojawi się okno końcowe, które zamykamy, klikając na **Close** D.

**Add Python 3.5 to PATH angażuje zmienną środowiskową Windows. Bez tego nasz system operacyjny nie będzie w stanie rozpoznać programów stworzonych przez środowisko IDLE i połączyć ich z tym środowiskiem.**

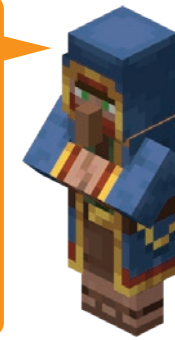
* We wskazówkach posłużono się nieco wcześniejszą wersją Pythona (zamiast najnowszej w momencie pisania książki wersji 3.8), by uniknąć ryzyka problemów z kompatybilnością z pozostałymi elementami opisywanego zestawu programów.

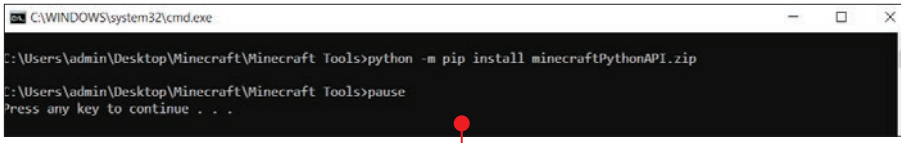
## API Minecrafta

API, czyli Interfejs Programistyczny Aplikacji, to łącznik między programem, który stworzymy, a światem Minecrafta. Pozwoli połączyć napisany przez nas skrypt z danymi pochodzącymi z plików gry Minecraft i w ten sposób uzyskać zamierzony przez nas efekt. Innymi słowy, API jest to pośrednik w komunikacji skryptu z grą.

Aby w naszym wypadku zainstalować API Minecrafta, musimy odnaleźć na dołączonej do książki płycie plik **Install_API.bat** (**Minecraft Tools** DVD-KOD: 004). Plik z rozszerzeniem .bat jest to program komputerowy wykonujący się w trybie wsadowym, to znaczy w sposób wykluczający ingerencję użytkownika w proces instalacyjny. Naszym zadaniem jest uruchomienie pliku, co wywoła pojawienie się konsoli CMD (Wiersza polecenia), który wykona instalację. Po jej zakończeniu wystarczy nacisnąć dowolny klawisz na klawiaturze komputera.

**API** można porównać do kelnera w restauracji. Gdy dostajemy menu, tworzymy zamówienie, które następnie przekazujemy kelnerowi. Kelner dostarcza je do kuchni, a ta przygotowuje zamówiony posiłek. Jeżeli jednak nie może przygotować dania, bo zabrakło na przykład pomidorów, to o tym problemie poinformuje nas właśnie kelner. W ten sposób działa API.

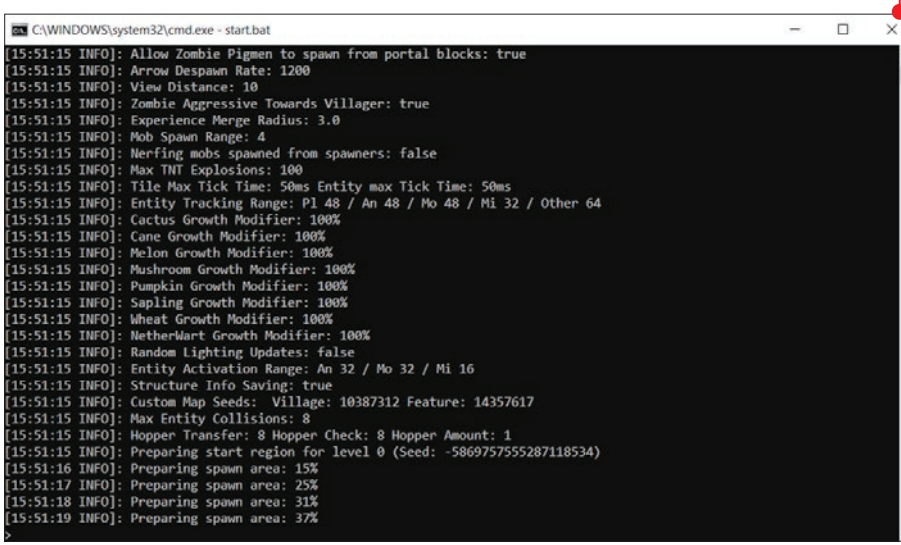





## Lokalny serwer Minecrafta

To ostatni element wymagający konfiguracji. Bez niego tworzone skrypty nie połączą się i nie wykonają w grze.

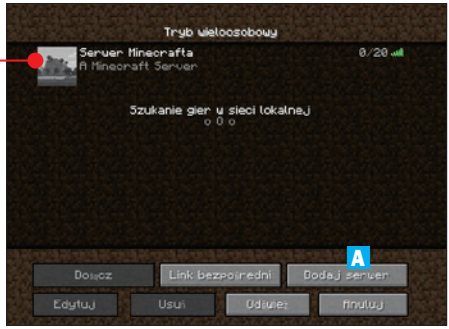
Serwer Minecrafta znajdziemy w plikach znajdujących się na płycie dołączonej do książki (**Minecraft Tools** DVD-KOD: 004). Odszukujemy **Start_Server** i uruchamiamy go dwukrotnym kliknięciem. Po wykonaniu tego kroku ponownie pojawi się znana już nam konsola CMD.



## UWAGA!

**W przypadku IDLE korzystamy z Minecraft Java Edition.**

Działanie naszego serwera należy rozpocząć po uruchomieniu gry Minecraft. Uruchamiamy grę Minecraft i lokalny serwer. Kolejne wskazówki w książce zostały przygotowane z wykorzystaniem wersji gry i serwera 1.9.4.



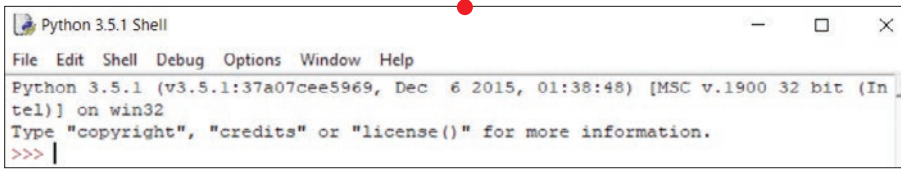

Gdy uruchomimy grę i serwer, wybieramy opcję **Tryb wieloosobowy**. Jeżeli prawidłowo uruchomiliśmy lokalny serwer, będzie on widoczny w formie wpisu na liście. Zielone kreski po prawej stronie oznaczają stabilność połączenia z serwerem.

Jeśli nie widzimy naszego serwera na liście, warto skorzystać z opcji dodawania ręcznego – klikamy na przycisk **Dodaj serwer** A. Dzięki temu będziemy mogli dostać się do naszego serwera od strony gry.
W polu **Adres serwera** B należy wpisać **localhost**. Klikamy na przycisk **Gotowe**. Gdy mamy już połączenie z serwerem, możemy rozpocząć programowanie Minecrafta przy użyciu języka Python.

## Rozpoczęcie pracy z IDLE

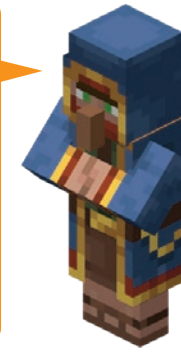
Uruchamiamy zainstalowany wcześniej program IDLE.
Musimy być przygotowani na to, że nie będziemy stosować tu żadnych bloków czy puzzli – będziemy tworzyć własny program tylko za pomocą różnych elementów tekstowych. Na początek wykonajmy bardzo prosty, niepowiązany z Minecraftem test – żeby wypróbować IDLE.
Klikamy na **File** u góry okna, a następnie na **New File**. Spróbujmy teraz wpisać w nowym oknie tekstowym programu matematyczne

obliczenie, na przykład **#2+2=4**. Aby przejść do kolejnej linijki, naciskamy enter.



Pierwsze działające programy będziemy testować w następnych częściach książki, gdzie poznamy wiele podstawowych pojęć programistycznych. Nasza misja będzie bardzo podobna do prawdziwej pracy programisty. Będziemy pisać kod, a potem go testować. Kod ten będzie tworzył nasz program.
Każde nowe pojęcie programistyczne postaramy się przetestować na odpowiednim programie.
Pamiętajmy, aby lokalny serwer był włączony cały czas, by uniknąć problemów z połączeniem.

**W języku programowania Python linijka kodu zaczynająca się od znaku # to komentarz – taka linijka się nie wykona. W komentarzach warto umieszczać informacje dotyczące kodu. Pamiętaj o dokładnym wpisywaniu kodu – drobna pomyłka może sprawić, że program nie będzie działał.**



# Teleportacja

Zaczniemy od poznania pojęcia **zmiennych** na podstawie teleportacji. Teleportacja to możliwość przemieszczania się z jednego miejsca na inne w mgnieniu oka. Postacią zdolną do naturalnej teleportacji w Minecrafcie jest Enderman. Aby modyfikować zjawisko teleportacji w Minecrafcie, musimy rozpocząć od poznania zmiennej.

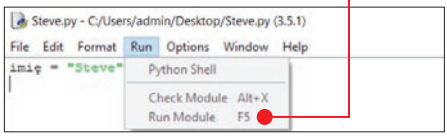
## Poznajemy zmienne

Zmienna w programowaniu to pewna wartość, która zmienia się w trakcie działania programu. Na przykład zmienną może być koordynata wysokości. Jeżeli teleportujemy się z poziomu 0 bloku na poziom 60 bloku, to wysokość położenia naszej postaci się zmieni. Na początku spróbujemy przetestować zmienne bez wprowadzania ich do Minecrafta.

**1** Otwieramy nowe okno, klikając na **File**, **New File**. Wprowadzimy teraz do konsoli pierwszą zmienną i wyświetlimy jej wartość.

```
imię = „Steve”
```

Po wpisaniu pierwszej linijki kodu klikamy na **Run** i **Run Module** ●.

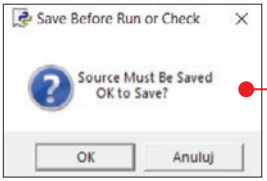



## BESTIARIUSZ – ENDERMAN

**Punkty życia:** 40
**Upuszcza:** Perła Endu
**Spawn:** poniżej 7 poziomu światła, w Endzie i w Netherze

**Charakterystyka:** Neutralny mob, którego możemy sprowokować poprzez patrzenie na niego oraz atakowanie go. Ma zdolności teleportacyjne – potrafi się teleportować w różne strony w promieniu 32 bloków. Nie lubi wody i unika jej.

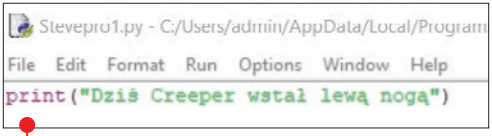
**Dodatkowe informacje:**
- w największej ilości występuje w Endzie
- potrafi pojawić się przy łóżku gracza, gdy ten śpi, i atakować go
- wydaje charakterystyczne dźwięki, które są zniekształconymi angielskimi słowami




**2** Zobaczymy informację, że trzeba zapisać projekt. Po zapisaniu projektu zostaniemy przeniesieni do konsoli o nieco zmienionej zawartości. Teraz wystarczy wpisać słowo „imię", a program sam napisze imię, które było zadeklarowane w poprzednim punkcie.

```
>>> imie
'Steve'
>>>
```

**3** Spróbujemy teraz wykorzystać funkcję **print**, która pozwala wyświetlić dowolny tekst na konsoli. Jest to również pierwsza komenda, którą będziemy wykorzystywać w testach.

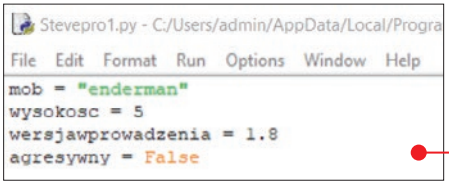



Wpisujemy komendę **print** i w nawiasie, z cudzysłowem, tekst, jaki ma zostać wyświetlony w programie. Następnie standardowo wywołujemy kod za pomocą **Run**, **Run Module**. Dzięki temu uzyskamy efekt w postaci tekstu wyświetlonego w pierwszym oknie.

W naszym przypadku zmienną, którą poznaliśmy, było imię, czyli zmienna typu tekstowego. Wpisywaliśmy ją, korzystając z cudzysłowu. Istnieją również inne typy zmiennych:

- **int** (liczba całkowita),
- **float** (liczba zmiennoprzecinkowa),
- **bool** (zmienna logiczna).

**Uruchomić kod można również za pomocą skrótu klawiaturowego – stosujemy do tego klawisz F5. Podobnie tworzenie nowego pliku tekstowego możemy wykonać skrótem klawiaturowym – w tym przypadku jest to kombinacja klawiszy Ctrl+N.**




**4** Teraz spróbujemy wprowadzić wszystkie typy zmiennych do skryptu:

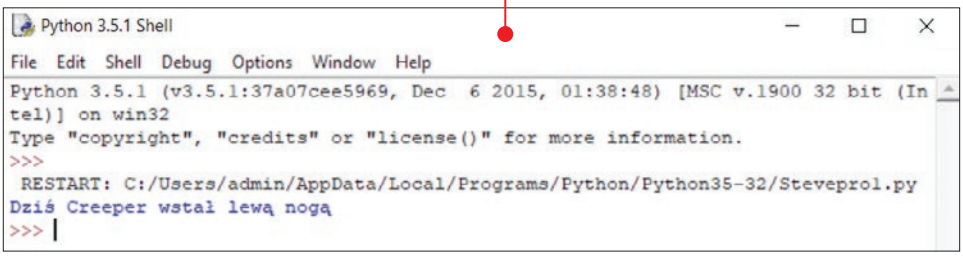
**mob = „enderman"**
**wysokosc = 5**
**wersjawprowadzenia = 1.8**
**agresywny = False**

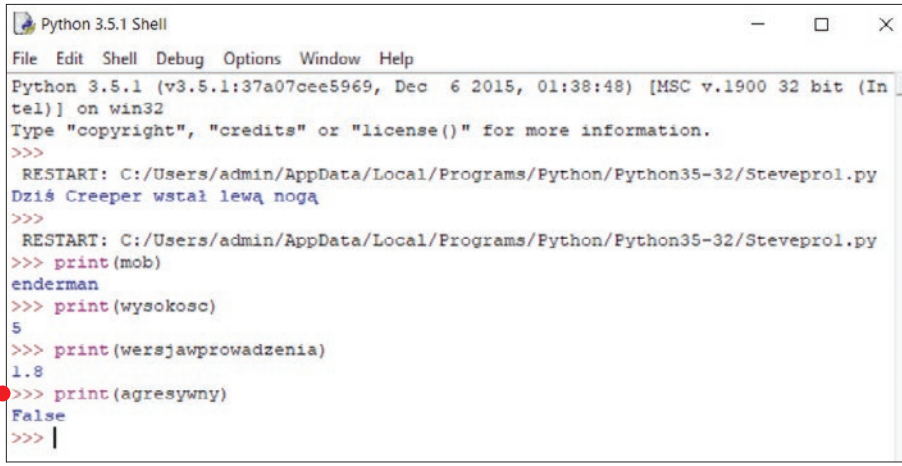
Każdą ze zmiennych możemy wyświetlić za pomocą funkcji **print**.

Jak widzimy, funkcja **print** sprawiła, że jesteśmy w stanie zobaczyć każdą ze zmiennych. Tworzenie zmiennych rządzi się pewnymi zasadami, których przestrzeganie jest konieczne, aby zmienne wyświetlały się i funkcjonowały prawidłowo.

Oto zasady tworzenia zmiennych:

- Zmienna powinna być zapisana z małej litery.
- Nazwa zmiennej nie powinna zawierać spacji.

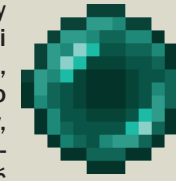





- Zmienna nie może zaczynać się od cyfry.
- Nazwa nie może zawierać elementów kluczowych (jak na przykład **print** – stworzenie zmiennej o nazwie takiej samej jak funkcja wywoływania tekstu wygenerowałoby błąd).
- Nazwa zmiennej powinna odnosić się do jej przeznaczenia i zachowania – na przykład jeżeli mamy zmienną, która ma wyświetlać wysokość, to najlepiej nazwać ją po prostu **wysokosc** lub podobnie. Wykorzystanie innej, nic niemówiącej nazwy na początku może się wydawać nieistotne, jednak gdy będziemy mieli kilkaset lub nawet kilka tysięcy linijek kodu, to takie niepowiązane z przeznaczeniem i działaniem nazwy zmiennych mogą być problemem.

Najwyższa pora wykorzystać wiedzę na temat zmiennych i spróbować swoich sił w zaprogramowaniu naszej postaci do teleportacji.

## Z RAPORTU STEVE'A



Przedmiotem, który umożliwia graczowi teleportację w grze, jest **Perła Kresu**. Po rzuceniu jej czekamy, aż uderzy w coś – wtedy też nasza postać jest teleportowana do danego miejsca. Perła Kresu pozwala również przejść przez granicę świata gry. Każde użycie Perły zabiera graczowi 5 punktów życia.

### Teleportacja bez Perły Kresu

W Minecrafcie jedną ze zmiennych, na których można pracować, są koordynaty. Koordynaty pozwalają zlokalizować naszą postać na mapie. Gdy wędrujemy lub wspinamy się, lokalizacja zmienia się o 1 blok. Szybkość zmiany zależy oczywiście od szybkości poruszania się i rodzaju transportu.

Co jednak, jeżeli lokalizacja zmieni się od razu na przykład o 20 bloków? Wtedy następuje teleportacja!

Zaprogramujemy teraz teleportację niezależną od wykorzystania Perły Kresu. Spróbujemy nadać naszej postaci takie właściwości, jakie ma Enderman. Zaczynamy oczywiście od przygotowania odpowiedniego kodu.

```
*Untitled*
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
```

**1** Do naszego pliku tekstowego wprowadzamy formułę ●.

**Ważne:** Każdy z naszych skryptów będziemy zaczynać od tych dwóch linijek tekstu. Są one niezbędne do ingerencji w świat Minecrafta.

```
*Untitled*
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
mc.player.setTilePos (25,70,50)
```

**2** Następnie wykorzystamy funkcję ustalenia lokalizacji gracza, czyli **mc.player.setTilePos** ●. Liczby w nawiasie po tej komendzie to koordynaty w Minecrafcie – pierwsza to X, druga – Y, a trzecia – Z.

**Jeżeli nie pamiętasz koordynat, to przypominam, że X i Z pozwalają zlokalizować postać na terenie mapy płaskiej, a koordynata Y umożliwia określenie wysokości względem świata.**

**3** Następnie zmieniamy nieco nasz kod – dzięki temu przekonamy się, że nie trzeba bezpośrednio wpisywać wartości X, Y, Z. Można je na początku zadeklarować A i potem się do nich odwołać. Za chwilę zobaczymy, jak to wygląda w programie.

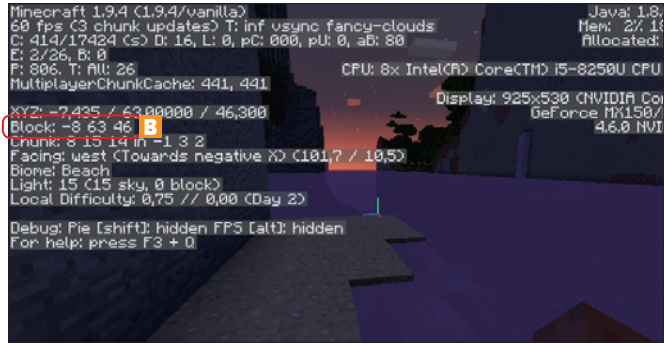
**4** Otwieramy grę w trybie wieloosobowym – zgodnie z instrukcją zamieszczoną wcześniej (pamiętajmy, że lokalny serwer musi być włączony) i uruchamiamy skrypt (na przykład naciskając klawisz F5). Co się stanie? Nasza postać teleportuje się w wybrane miejsce, jednak dzieje się to automatycznie, przez co nawet nie jesteśmy w stanie dobrze uchwycić wzrokiem tego momentu.

```
Test1.py - C:/Users/admin/AppData/Local/Programs/Pyt
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
x = 10
y = 110
z = 12
mc.player.setTilePos (x, y, z)
```

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
import time
time.sleep(5)
x = 10
y = 110
z = 12
mc.player.setTilePos (x, y, z)
```

**5** Dlatego warto do naszego skryptu dodać opóźnienie, które sprawi, że teleportacja odbędzie się dopiero po chwili. Aby to zrobić, wprowadzamy do kodu następujące elementy ●. Dzięki temu możemy zacząć wędrować po mapie, a dopiero po chwili zostaniemy teleportowani w miejsce wyznaczone przez koordynaty.

**Ważne:** Aby przetestować działanie skryptu, warto podczas gry mieć włączone wyświet-



lanie ukrytych danych, takich jak aktualna lokalizacja gracza. Dane te włączamy, w grze przy użyciu klawisza F3.

Jak widzimy, po rozpoczęciu gry postać Steve'a znajduje się w lokalizacji **-8**, **63**, **46** B. Koordynaty po teleportacji to **10**, **105**, **12** C. Zgadzają się koordynaty X i Z natomiast Y jest o 5 bloków niższe. Dzieje się tak dlatego, że zostaliśmy teleportowani na odpowiednią wysokość, ale niestety znajdowało się tam tylko powietrze, więc postać zaczęła spadać do czasu, aż trafiła na blok inny niż powietrze.

**6** Spróbujmy teraz zrobić ciąg teleportacyjny, wykorzystując poznane komendy.

Test1.py - C:/Users/admin/AppData/Local/Programs/Pyt

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
import time
time.sleep(5)
mc.player.setTilePos (100, 100, 100)
time.sleep(1)
mc.player.setTilePos (90, 100, 90)
time.sleep(1)
mc.player.setTilePos (80, 100, 80)
time.sleep(1)
mc.player.setTilePos (70, 100, 70)
time.sleep(1)
mc.player.setTilePos (60, 100, 60)
time.sleep(1)
```

**Przykład ciągu teleportacyjnego**

Pamiętajmy, aby wykorzystywać funkcję **time.sleep**, dzięki której między jedną teleportacją a drugą będzie chwila przerwy.

**7** A teraz zaprogramujemy inny typ teleportacji, w którym gra wykorzysta obecną lokalizację gracza i na tej podstawie doda do niej jakąś liczbę, teleportując jego postać.

*Test1.py - C:/Users/admin/AppData/Local/Programs/

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
import time
time.sleep(5)
pozycja = mc.player.getTilePos()
```

Aby to zrobić, otwieramy nowe okno konsoli i wprowadzamy funkcję badającą obecne koordynaty postaci gracza.

**Ważne:** Opóźnienie pozostało na swoim miejscu, aby teleportacja nie odbywała się natychmiastowo wraz ze startem gry. Następnie zmieniamy wartości X, Y, Z zgodnie z przykładem.

*Test1.py - C:/Users/admin/AppData/Local/Programs/

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
import time
time.sleep(5)
pozycja = mc.player.getTilePos()
x = pozycja.x
y = pozycja.y
z = pozycja.z
```

**8** W tym momencie jeszcze nic się nie stanie ze względu na to, że w zasadzie nigdzie nie została zmieniona pozycja. Spróbujmy więc do jednej z pozycji dodać dowolną liczbę i sprawdźmy, co się wydarzy. W konsekwencji postać gracza przeskoczy z jednego miejsca na drugie.

Warto samodzielnie wypróbować różne elementy teleportacji!

*Test1.py - C:/Users/admin/AppData/Local/Programs/Python/Python35-
File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 20
y = pozycja.y
z = pozycja.z

mc.player.setTilePos (x, y, z)
```

# Miniprojekt Skok na bungee

Sprawdzimy teraz, czy opanowaliśmy teleportację. Wykorzystamy pewną minigrę opartą na szczęściu. Polega ona na teleportowaniu gracza na maksymalną dostępną wysokość. Gracz zaczyna spadać. Jeżeli uda nam się teleportować gracza nad wodę, to wygrywamy, bo nasz bohater przeżyje skok. Jeśli jednak upadniemy na twarde bloki, to, niestety, przegrywamy. Projekt ten oparty jest właśnie na sprawdzaniu różnych koordynatów, dzięki czemu będzie idealny do testowania naszej wiedzy o teleportacji.

*Test1.py - C:/Users/admin/AppData/Local/Programs/
File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
```

**1** Zaczynamy oczywiście od niezbędnych elementów umożliwiających pracę z Minecraftem **A**. Następnie dodajemy funkcję oczekiwania na skok **B**.

*Test1.py - C:/Users/admin/AppData/Local/Programs/Pythc
File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(10)
```

A

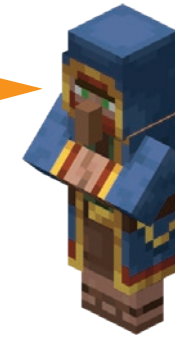
B

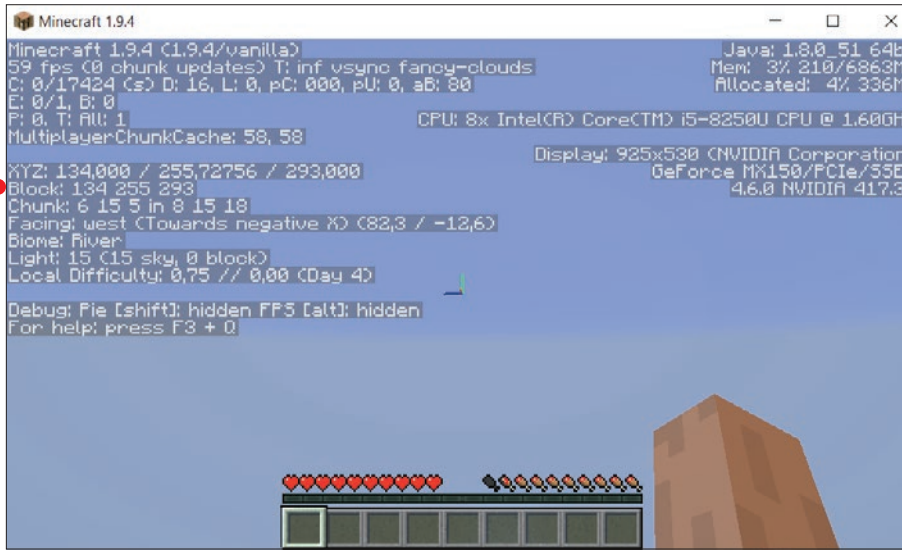
*Test1.py - C:/Users/admin/AppData/Local/Programs/
File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(10)

x = 134
y = 256
z = 293
```

**2** Kolejnym etapem jest wprowadzenie zmiennych – koordynat X, Y i Z.

**Wartość Y musi być stała podczas naszych testów ze względu na to, że jest to maksymalna wysokość dostępna w Minecrafcie. Inne koordynaty możesz zmieniać dowolnie.**






**3** Ostatnim etapem jest teleportacja do wyznaczonych koordynat.

**4** Sprawdźmy, czy kod działa, uruchamiając go i przechodząc ponownie do gry. Po przetestowaniu sprawdzamy też inne koordynaty, pozostawiając jednak koordynatę Y niezmienioną, aby postać mogła spadać. Zmieniajmy wartości X i Z o wiele bloków, aby trafiać na inne biomy. Unikajmy sytuacji, gdy w pierwszym skoku będą to wartości takie jak 1 i 1, a w drugim 2 i 2, bo wtedy wylądujemy na podobnym terenie.

*Test1.py - C:/Users/admin/AppData/Local/Programs/Pyth

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(10)

x = 134
y = 256
z = 293

mc.player.setTilePos (x, y, z)
```

# Tworzenie bloków, brył i konstrukcji

Bloki to podstawa funkcjonowania Minecrafta. Cały świat jest stworzony z bloków. Również powietrze i woda to bloki, mimo że wyglądają, jakby stanowiły spójną całość.

## Poznajemy funkcję tworzącą bloki

Aby przy użyciu Pythona dodać blok w Minecrafcie, należy skorzystać z funkcji **setBlock**:

**mc.setBlock(x,y,z,typ)**

Funkcja ta, jak widzimy, również ma koordynaty X, Y, Z, które oznaczają miejsce, gdzie

### Z RAPORTU STEVE'A

Bloki są bazową jednostką budulcową Minecrafta. Powierzchnię 16 bloków wzdłuż i wszerz oraz 256 bloków wysokości nazywamy chunkiem. Naciskając skrót klawiaturowy F3+G, możemy wyświetlić granice chunków.

dany blok zostanie postawiony. Natomiast typ to rodzaj bloku, jaki chcemy postawić. Określając położenie oraz podając ID wybranego bloku (patrz tabela na stronie obok), napiszemy kod tworzący przykładowy blok.

## Tworzymy blok gąbki

**1** Zaczynamy od standardowych elementów wraz z opóźnieniem czasowym.

```
Test1.py - C:/Users/admin/AppData/Local/Programs/P
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)
```

**2** Następnie dodajemy funkcję sprawdzającą pozycję gracza. Jest to konieczne, aby blok postawić niedaleko niego i móc zobaczyć efekt działania kodu.

```
Test1.py - C:/Users/admin/AppData/Local/Programs/P
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()
```

**3** Kolejnym etapem jest dodanie zmiennych X, Y, Z – to obecna pozycja gracza. **+1** przy X to postawienie danego bloku obok gracza.

```
Test1.py - C:/Users/admin/AppData/Local/Programs/P
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
```

**Typ bloku oznaczamy numerycznie. Jest to ID bloku, czyli jego numer identyfikacyjny. Nie tylko bloki mają swoje ID, ale także moby, przedmioty, biomy i inne. Listę ID najpopularniejszych bloków znajdziesz na stronie obok.**

## Z RAPORTU STEVE'A

**Gąbka to blok, który potrafi wchłaniać wodę. Przydaje się do osuszania kopalni. Może wchłonąć 65 bloków wody (w promieniu 7 bloków od miejsca jej postawienia). W sposób naturalny występuje w podwodnych świątyniach.**

**4** Potem musimy dodać ID bloku, jaki chcemy postawić, na przykład niech to będzie gąbka (**19** to numer identyfikacyjny gąbki).

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 19
```

**5** Ostatnim elementem jest wywołanie funkcji **setBlock** A (patrz strona 86), która postawi wybrany blok w określonej przez nas wcześniej lokalizacji.

# LISTA ID NAJPOPULARNIEJSZYCH BLOKÓW

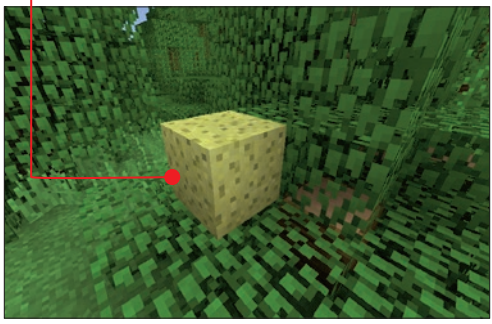
| BLOK | ID | OPIS |
|---|---|---|
| | 01 | Kamień |
| | 02 | Blok trawy |
| | 03 | Ziemia |
| | 04 | Bruk |
| | 05 | Deski |
| | 07 | Skała macierzysta |
| | 08 | Woda |
| | 10 | Lawa |
| | 12 | Piasek |
| | 13 | Żwir |
| | 14 | Ruda złota |
| | 15 | Ruda żelaza |
| | 16 | Ruda węgla |
| | 17 | Drewno |
| | 18 | Liście |
| | 19 | Gąbka |
| | 20 | Szkło |

| BLOK | ID | OPIS |
|---|---|---|
| | 21 | Ruda lazurytu |
| | 22 | Blok lazurytu |
| | 41 | Blok złota |
| | 42 | Blok żelaza |
| | 45 | Cegły |
| | 46 | TNT |
| | 47 | Biblioteczka |
| | 48 | Zamszony bruk |
| | 49 | Obsydian |
| | 56 | Ruda diamentu |
| | 57 | Blok diamentu |
| | 58 | Stół rzemieślniczy |
| | 73 | Ruda czerwonego kamienia |
| | 86 | Dynia |
| | 87 | Netherrack |
| | 88 | Piasek dusz |
| | 89 | Jasnogłaz |

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 19

mc.setBlock(x,y,z,typ)
```

A

**6** Wypróbujmy skrypt, naciskając klawisz F5. Przechodzimy od razu do świata Minecrafta. Po uruchomieniu skryptu powinniśmy zobaczyć blok gąbki znajdujący się w odległości jednego bloku od naszej postaci.



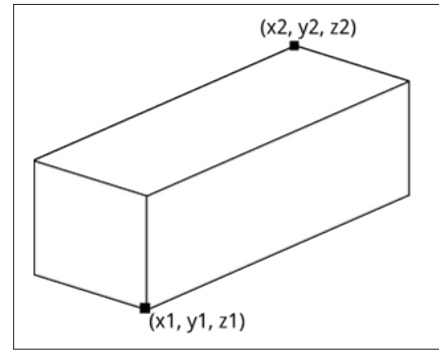
## UWAGA!

**Pamiętaj, aby podczas programowania gra i lokalny serwer cały czas były uruchomione w tle.**

### Tworzymy bryłę z bloków dynamitu

Często zdarza się jednak, że jeden blok to za mało i chcemy stworzyć większą konstrukcję. Układanie bloków jeden po drugim zajmuje jednak zbyt dużo czasu. Na szczęście programiści przygotowali się na taki scenariusz, dzięki czemu mamy funkcję **setBlocks**, która umieszcza kilka bloków w określonym zakresie lokalizacji:

**mc.setBlocks(x1,y1,z1,x2,y2,z2,typ)**




Aby ułatwić zrozumienie tych elementów, koordynaty tego samego typu zostały oznaczone kolorami i umieszczone na schemacie prostopadłościanu. Jak widzimy, X1, Y1, Z1 to jest prawy dolny róg naszej konstrukcji, natomiast X2, Y2 i Z2 to lewy górny róg. Dzięki oznaczeniu koordynat w grze otrzymamy daną konstrukcję.

## UWAGA!

**Uważaj, aby nie przesadzić z wielkością konstrukcji. Czasami zdarza się, że gra się zawiesza lub całkowicie się wyłącza, jeżeli konstrukcja jest za duża.**

## Z RAPORTU STEVE'A

**TNT to ładunek wybuchowy wytwarzany z prochu i piasku (lub czerwonego piasku). Czas wybuchu trwa 4 sekundy. Aby TNT wybuchło, należy je wcześniej aktywować, na przykład za pomocą krzesiwa albo innego elementu zapalającego (lawa, płonąca strzała, czerwona pochodnia itp.).**

Spróbujemy teraz stworzyć podobną konstrukcję. Blokiem, który będzie budował naszą bryłę, będzie TNT.

**1** W celu stworzenia bryły dynamitu zastosujemy część skryptu napisaną w poprzednim poradniku o bloku gąbki. Dzięki temu będziemy mogli szybciej wykorzystać funkcję **setBlocks**.

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/P
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 46
```

**2** Pamiętamy w tym miejscu o zmianie ID bloku. Poprzednio korzystaliśmy z gąbki o ID 19. TNT ma numer identyfikacyjny **46**, więc taki numer należy wprowadzić do naszego skryptu.

**3** Kolejnym etapem będzie dodanie zmiennych szerokość, wysokość, długość,

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 46


szer = 10
wys = 10
dl = 10
```

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/Python/Python35
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time

time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 46


szer = 10
wys = 10
dl = 10
mc.setBlocks(x,y,z,x + szer, y + wys, z + dl, typ)
```

które pozwolą nam na określenie koordynatów X2, Y2 i Z2.

**4** W tym momencie jesteśmy gotowi na wykorzystanie funkcji **setBlocks**. Koordynaty **x1**, **y1** i **z1** będą koordynatami położenia naszej postaci (X1 będzie przesunięty o 1 blok, gdyż w kodzie mamy zapisane **x= pozycja.x + 1**), natomiast koordynaty **x2**, **y2**, **z2** powstaną poprzez dodanie określonej szerokości, wysokości i długości do koordynat **x1**, **y1** i **z1**.

**5** Uruchamiamy skrypt i sprawdzamy, czy udało się stworzyć wielki blok dynamitu.

**6** Przed przejściem do następnego projektu wypróbujmy jeszcze poniższy kod. Pozwala on na stworzenie bloku, który będzie miał pustą przestrzeń w środku.

*Test1.py - C:/Users/admin/AppData/Local/Programs/Python/Python35-32/Test1.py (3.5.1)*

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time
time.sleep(5)

pozycja = mc.player.getTilePos()
x = pozycja.x
y = pozycja.y
z = pozycja.z

blokZiemia = 3

szerokosc = 10
wysokosc = 10
dlugosc = 10

blokPowietrze = 0


mc.setBlocks(x+1,y,z+1,x+szerokosc,y+wysokosc,z+dlugosc,blokZiemia)
mc.setBlocks(x+2,y+1,z+2,x+szerokosc-1,y+wysokosc-1,z+dlugosc-1,blokPowietrze)
```

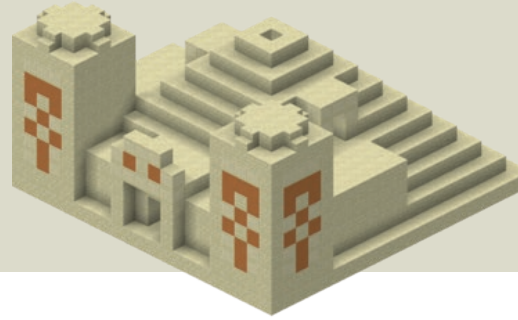
# Miniprojekt Piramida

Wykorzystamy teraz umiejętność tworzenia bloków i stawiania ich w określonych miejscach do zbudowania małej piramidy. Będzie to piramida na kształt egipski, a nie taka, jaka powszechnie występuje w Minecrafcie.
Przygotujmy się od razu na to, że tworzenie piramidy będzie dość powtarzalnym procesem (każda warstwa będzie budowana w podobny sposób), więc musimy wykazać się cierpliwością.

**1** Zaczynamy od standardowych elementów **A**, do których będziemy po kolei dołączać następne komendy.

## Z RAPORTU STEVE'A

**Piramida to struktura naturalnie generująca się w biomach pustynnych. Znajdują się w niej cztery skrzynie. Należy jednak uważać – wewnątrz znajduje się pułapka w postaci płyty naciskowej. Jeżeli na nią nadepniemy (lub zrobi to jakiś mob), to aktywujemy TNT znajdujące się pod podłogą, wywołując wybuch.**



*Test1.py - C:/Users/admin/AppData/Local/Programs/

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time
time.sleep(5)
```

A

**2** Następnie ustalamy lokalizację i typ bloku, z jakiego będziemy korzystać. Nasza piramida zbudowana będzie z piasku, a więc numer identyfikacyjny to **12**.

*Test1.py - C:/Users/admin/AppData/Local/Programs/Pytho

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time
time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 12
```

**3** Kolejnym etapem będzie wywołanie funkcji **setBlocks**. (Będziemy za chwilę wywoływać tę funkcję kilkakrotnie dla każdej warstwy piramidy). Wykonanie kodu w tym momencie sprawi, że pojawi się pierwsza warstwa piasku.

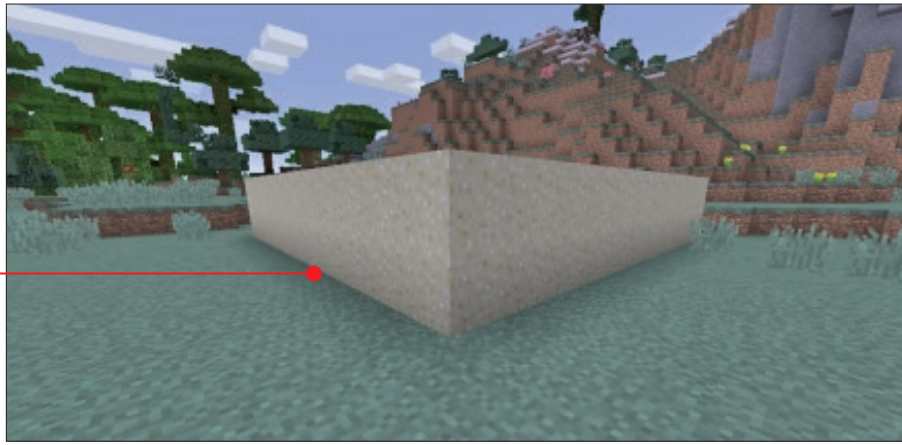
Test1.py - C:/Users/admin/AppData/Local/Programs/Python/Pytho

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time
time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 12

mc.setBlocks(x,y ,z,x + 10, y + 1, z + 10, typ)
```

## UWAGA!

Pamiętaj, że piasek należy do bloków, na które działa w Minecrafcie grawitacja. Znaczy to, że gdy umieścimy go w miejscu, w którym nie będzie miał solidnych bloków pod sobą, to będzie spadał do momentu, aż na czymś się oprze. Postaraj się więc tworzyć swoją piramidę na płaskim terenie.



**4** Dodajemy teraz drugą warstwę piramidy A.

**5** Potem dodajemy kolejno następne warstwy B.

**6** W efekcie, jak możemy sprawdzić, otrzymujemy małą piramidę.

**7** Stosując tę samą metodę warstw,

*Test1.py - C:/Users/admin/AppData/Local/Programs/Python/Python35-32/Test1.py (3.5.1

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time
time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 12

mc.setBlocks(x,y ,z,x + 10, y + 1, z + 10, typ)
mc.setBlocks(x + 1,y + 1,z + 1 ,x + 9, y + 1, z + 9, typ)
```

A

Test1.py - C:/Users/admin/AppData/Local/Programs/Python/Python35-32/Test1.py (3.5.1)

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time
time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 12

mc.setBlocks(x,y ,z,x + 10, y + 1, z + 10, typ)
mc.setBlocks(x + 1,y + 1,z + 1 ,x + 9, y + 1, z + 9, typ)
mc.setBlocks(x + 2,y + 2,z + 2 ,x + 8, y + 2, z + 8, typ)
mc.setBlocks(x + 3,y + 3,z + 3 ,x + 7, y + 3, z + 7, typ)
mc.setBlocks(x + 4,y + 4,z + 4 ,x + 6, y + 4, z + 6, typ)
mc.setBlocks(x +5 ,y + 5,z + 5 ,x + 5, y + 5, z + 5, typ)
```

B

Test1.py - C:/Users/admin/AppData/Local/Programs/Python/Python35-32/Test1.py (3.5.1)

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

import time
time.sleep(5)

pozycja = mc.player.getTilePos()

x = pozycja.x + 1
y = pozycja.y
z = pozycja.z
typ = 41

mc.setBlocks(x + 5,y ,z + 5 ,x + 5, y, z + 5, typ)
mc.setBlocks(x + 4,y + 1,z + 4 ,x + 6, y + 1, z + 6, typ)
mc.setBlocks(x + 3,y + 2,z + 3 ,x + 7, y + 2, z + 7, typ)
mc.setBlocks(x + 3,y + 3,z + 2 ,x + 8, y + 3, z + 8, typ)
mc.setBlocks(x + 1,y + 4,z + 1 ,x + 9, y + 4, z + 9, typ)
mc.setBlocks(x, y + 5, z,x + 10, y + 5, z + 10, typ)
mc.setBlocks(x + 1,y + 6,z + 1 ,x + 9, y + 6, z + 9, typ)
mc.setBlocks(x + 2,y + 7,z + 2 ,x + 8, y + 7, z + 8, typ)
mc.setBlocks(x + 3,y + 8,z + 3 ,x + 7, y + 8, z + 7, typ)
mc.setBlocks(x + 4,y + 9,z + 4 ,x + 6, y + 9, z + 6, typ)
mc.setBlocks(x + 5,y + 10,z + 5 ,x + 5, y + 10, z + 5, typ)
```

spróbujmy wykorzystać poznany skrypt, aby stworzyć odwróconą klepsydrę. Zwróćmy uwagę na ID bloku użytego w przykładzie ●. Nasza odwrócona klepsydra zrobiona jest tym razem ze złota.

**8** Uruchamiając skrypt ●, zobaczymy, że powstały dwie piramidy złączone podstawami ●.

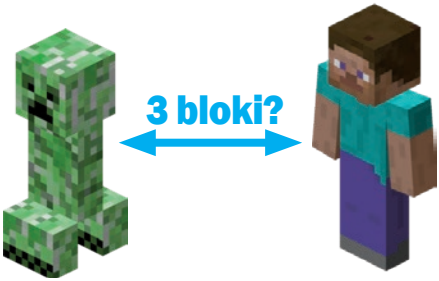
**Jeśli chodzi o bloki, to na razie już wszystko. Pora przejść do bardziej zaawansowanych zagadnień programistycznych.**

# Prawda/Fałsz

Bardzo często używane są zmienne typu logicznego. Są to zmienne oparte tylko na dwóch elementach – prawdzie lub fałszu. Zmienne logiczne najłatwiej zobrazować przykładem włącznika światła. Wyobraźmy sobie, że jeżeli światło jest włączone, to zmienna logiczna przyjmuje wartość Prawda (**True**), a jeżeli światło jest wyłączone, to zmienna przyjmuje wartość Fałsz (**False**). Zmienne logiczne można wykorzystać, na przykład sprawdzając, czy dany warunek jest spełniony. Sprawdzanie warunków odbywa się za pomocą operatorów. W Pythonie używa się sześciu operatorów:

**równa się (==)**
**nie równa się (!=)**
**mniejszy niż (<)**
**mniejszy bądź równy (<=)**
**większy niż (>)**
**większy bądź równy (>=)**

Zestawienie operatora ze zmiennymi pozwala uzyskać zmienną logiczną. Na przykład: Czy odległość gracza od Creepera wynosi 3 bloki?




**Co się dzieje w przypadku, gdy warunek zostaje spełniony, czyli odległość==3 bloki?**
Otrzymujemy zmienną logiczną o wartości **True**, a to inicjuje wybuch Creepera.

**Co się dzieje w przypadku, gdy warunek nie zostaje spełniony, czyli odległość!=3 bloki?**
Otrzymujemy zmienną logiczną o wartości **False**, a to sprawia, że Creeper stara się podejść bliżej.
Sprawdźmy, jak zmienne **True** i **False** wyglądają w świecie Minecrafta i jakie mają zastosowanie. Zaczniemy od wyłączenia destruktywności bloków.

```
Test1.py - C:/Users/admin/AppData/Local/Programs/P
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

mc.setting("world_immutable", True)
```

Po uruchomieniu tego skryptu spróbuj coś zniszczyć. Niestety, nie będzie to możliwe. Dopóki **world_immutable** ma wartość zmiennej logicznej ustawioną na **True**, świat będzie niezniszczalny. Spróbujmy odwrócić ten proces. Zamiast wartości zmiennej logicznej True wpisujemy **False**.

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/P
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

mc.setting("world_immutable", False)
```

Okazuje się, że wróciliśmy do stanu, w którym świat znów jest w standardowym trybie, to znaczy można go modyfikować i niszczyć bloki.
Kolejnym skryptem, który warto przetestować, jest skrypt sprawdzający, czy gracz stoi na suchym lądzie. Do tego celu wykorzystamy dwie nowe funkcje:

**mc.getBlock(x,y,z)**
oraz
**mc.postToChat**

Rozpoczynamy od standardowych elementów i określenia pozycji gracza.

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

pozycja= mc.player.getPos()
x= pozycja.x
y= pozycja.y
z= pozycja.z
```

Potem wprowadzamy komendę **getBlock**, która będzie odpowiedzialna za sprawdzenie, w jakim bloku się znajdujemy (można powiedzieć, że jest to komenda czujnik).

```
*Test1.py - C:/Users/admin/AppData/Local/Programs/Pyth
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

pozycja= mc.player.getPos()
x= pozycja.x
y= pozycja.y
z= pozycja.z

typBloku=mc.getBlock(x,y,z)
```

Kolejnym elementem będzie wyświetlenie na czacie w grze wyniku porównania za pomocą funkcji **postToChat**. Sprawi ona, że po każdym sprawdzeniu bloku zostanie wysłana wiadomość informująca nas, czy rzeczywiście stoimy na stałym bloku, czy też znajdujemy się na przykład w wodzie.
Wyrażenie **typBloku==0** w ostatniej linijce kodu sprawdza, czy blok, w którym się znajdujemy, to powietrze. Jeśli zwracana jest wartość **True**, oznacza, że gracz stoi w świecie Minecrafta (postać Steve'a znajduje się w bloku powietrza). Jeżeli zwracana jest wartość **False**, gracz jest w wodzie.

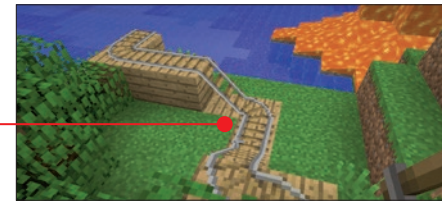
```
*Test1.py - C:/Users/admin/AppData/Local/Programs/Pyt
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc = Minecraft.create()

pozycja= mc.player.getPos()
x= pozycja.x
y= pozycja.y
z= pozycja.z

typBloku=mc.getBlock(x,y,z)

mc.postToChat(typBloku==0)
```

Gdy podczas uruchomienia skryptu nasza postać będzie pływać w wodzie, otrzymamy wartość False

Gdy w momencie uruchomienia skryptu znajdujemy się na lądzie, otrzymamy na czacie wartość True

# Miniprojekt Tor

Spróbujmy teraz wykonać projekt – tor dla wagonika. Tor będzie się pojawiać za graczem przez cały czas, dopóki on będzie się znajdować na suchym lądzie.



Test1.py - C:/Users/admin/AppData/Local/Programs/P

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc=Minecraft.create()
```

**1** Zaczynamy oczywiście od standardowych komponentów naszego kodu.

**2** Następnie importujemy czas oraz dodajemy funkcję:

**while True:**,

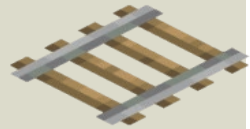
która oznacza **dopóki prawda** – elementy umieszczone w tej funkcji będą się wykonywać tak długo, jak warunek będzie przyjmował wartość logiczną **True**.

Test1.py - C:/Users/admin/AppData/Local/Programs/P

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc=Minecraft.create()

import time
while True:
```

## Z RAPORTU STEVE'A



Tor służy przede wszystkim do szybkiego transportu materiałów. Naturalnie generuje się w kopalniach. Dwa tory położone obok siebie będą się automatycznie łączyć. Ciekawostką jest to, że moby unikają przechodzenia przez tory.

**3** Potem pobieramy pozycję gracza i ustalamy typ bloku i toru, jaki będzie na nim umieszczany.

*Test1.py - C:/Users/admin/AppData/Local/Programs/P

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc=Minecraft.create()

import time
while True:
    pozycja=mc.player.getTilePos()
    x=pozycja.x
    y=pozycja.y
    z=pozycja.z
    typBloku=5
    tor=66
```

**4** Dodajemy drobne opóźnienie, które będzie się znajdować między jednym ułożeniem bloku i toru a drugim, oraz wprowadzamy funkcje **setBlock**.

*Test1.py - C:/Users/admin/AppData/Local/Programs/Py

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc=Minecraft.create()

import time
while True:
    pozycja=mc.player.getTilePos()
    x=pozycja.x
    y=pozycja.y
    z=pozycja.z
    typBloku=5
    tor=66
    time.sleep(0.1)
    mc.setBlock(x,y-1,z,typBloku)
    mc.setBlock(x,y,z,tor)
```

**5** W efekcie, chodząc po lądzie, będziemy umieszczać pod sobą blok drewna, a na nim tor.

## PĘTLE

Pętle w programowaniu pełnią wiele różnych funkcji. Są odpowiedzialne za prawidłową organizację programu i ułatwiają wiele czynności. Ich właściwym zadaniem jest cykliczne powtarzanie jakiegoś elementu w nieskończoność albo do czasu zajścia odpowiednich warunków lub określoną liczbę razy. W naszym programowaniu będziemy stosować pętlę **for**. Użyjemy jej w dwóch miniprojektach, które pokażą nam zasadę działania pętli w języku programowania Python. Pierwszy z nich będzie opierał się na poznanej wcześniej teleportacji, drugi to będzie zapętlone umieszczanie bloków. Pętle w tych wypadkach będą pętlami ograniczonymi liczbą powtórzeń – wykonają się określoną liczbę razy. Istnieją też oczywiście pętle nieograniczone, czyli takie, które wykonują się od momentu rozpoczęcia programu do jego zakończenia.

W języku programowania Python występują dwa rodzaje pętli: for (czyli ta, z której będziemy korzystać) oraz while – pętla wykonująca się tylko wtedy, gdy jakiś warunek jest spełniony, na przykład gdy gracz nie ma pod sobą solidnego bloku, spada, dopóki nie natknie się na solidny blok.

# Miniprojekt Zapętlona teleportacja

Projekt zapętlonej teleportacji będzie polegać na wykonaniu czterech pętli, które sprawią, że będziemy teleportować się kilkunastokrotnie, a następnie wrócimy na miejsce startu. Jeżeli martwisz się, że podczas teleportacji będą pojawiać się Endermite'y, to nie ma potrzeby – to jest inny rodzaj teleportacji niż ten, który oferuje Perła Kresu.

```
Pythontest.py - C:/Users/admin/AppData/Local/Program
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc= Minecraft.create()
import time
```

**1** Rozpoczynamy pracę od zaimportowania modułów obsługujących Minecrafta oraz importujemy czas.

```
Pythontest.py - C:/Users/admin/AppData/Local/Program
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc= Minecraft.create()
import time
time.sleep(5)
```

## BESTIARIUSZ – ENDERMITE

**Punkty życia:** 8
**Upuszcza:** brak
**Spawn:** 5% szansy po użyciu przez gracza Perły Kresu.

**Charakterystyka:**
Agresywny mob o niewielkich rozmiarach koloru ciemnofioletowego. Odporny na działanie wody. Znika po mniej więcej 2 minutach od pojawienia się

**Dodatkowe informacje:**
- otrzymuje obrażenia, gdy wędruje po Piasku Dusz
- atakowany przez Endermany

**2** Następnie dodajemy krótkie opóźnienie, aby proces teleportacji nie wykonywał się bezpośrednio po włączeniu gry.

**3** Ostatnim podstawowym etapem jest pobranie obecnej pozycji gracza w świecie za pomocą funkcji **mc.player.getPos** oraz przypisanie zmiennych do koordynat.

```
Pythontest.py - C:/Users/admin/AppData/Local/Programs
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc= Minecraft.create()
import time
time.sleep(5)
poz=mc.player.getPos()

x=poz.x
y=poz.y
z=poz.z
```

**4** Podstawowe elementy kodu już mamy. Teraz będziemy konstruować pętle, które będą odpowiedzialne za teleportację. Rozpoczynamy od przygotowania konstrukcji pierwszej pętli.

```
Pythontest.py - C:/Users/admin/AppData/Local/Program
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc= Minecraft.create()
import time
time.sleep(5)
poz=mc.player.getPos()

x=poz.x
y=poz.y
z=poz.z
for przod in range (1,11):
        time.sleep(0.1)
        mc.player.setPos(x+przod,y,z)
```

Zapis ten informuje nas, że pętla w zakresie 10 bloków przeniesie nas w przód. Co każdy blok następować będzie przerwa 0,1 s. Pętla pozwoli nam poruszyć się **do przodu**. W podobny sposób będziemy konstruować kolejne pętle:

### ruch w lewą stronę

```
for lewo in range (1,11):
    time.sleep(0.1)
    mc.player.setPos(x+10,y,z-lewo)
```

### ruch do tyłu

```
for tyl in range (1,11):
    time.sleep(0.1)
    mc.player.setPos(x+10-tyl,y,z-10)
```

### ruch w prawą stronę

```
for prawo in range (1,11):
    time.sleep(0.1)
    mc.player.setPos(x,y,z-10+prawo)
```

**5** Ostatecznie cały skrypt będzie wyglądał następująco:

```
Pythontest.py - C:/Users/admin/AppData/Local/Programs
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc= Minecraft.create()
import time
time.sleep(5)
poz=mc.player.getPos()

x=poz.x
y=poz.y
z=poz.z
for przod in range (1,11):
    time.sleep(0.1)
    mc.player.setPos(x+przod,y,z)
for lewo in range (1,11):
    time.sleep(0.1)
    mc.player.setPos(x+10,y,z-lewo)
for tyl in range (1,11):
    time.sleep(0.1)
    mc.player.setPos(x+10-tyl,y,z-10)
for prawo in range (1,11):
    time.sleep(0.1)
    mc.player.setPos(x,y,z-10+prawo)
```

**6** Po zapisaniu projektu uruchamiamy świat Minecrafta oraz lokalny serwer i testujemy naszą teleportację. Uważajmy jednak na to, w jakim miejscu świata testujemy teleportację. Najlepiej wykonywać ją na płaskim terenie, bez wzgórz czy dolin.

## Z RAPORTU STEVE'A

Po teleportowaniu się w solidny blok zaczynamy otrzymywać obrażenia od duszenia się. Najczęściej taka sytuacja ma miejsce w momencie, gdy nie załaduje nam się chunk i wejdziemy w pustą przestrzeń po nim. Chunk następnie się załadowuje i zostajemy uwięzieni w solidnych blokach.

# Miniprojekt Spiralne schody

W Minecrafcie jest mnóstwo różnego rodzaju konstrukcji. Część z nich jest imponujących, gigantycznych rozmiarów. Bardzo często takie budowle nie są wykonywane ręcznie, tylko są efektem pracy różnych programów, napisanych w języku Python, ale nie tylko. Tworzenie wielkiej konstrukcji mogłoby zająć nawet i kilka miesięcy, jeżeli chcielibyśmy robić wszystko ręcznie, blok po bloku. W tym poradniku skupimy się więc na wykorzystaniu języka Python i funkcji pętli do stworzenia prostej struktury schodów. Schody będą spiralne i pozwolą się wspiąć bardzo wysoko.

Spróbujemy uzyskać taki efekt poprzez użycie czterech pętli umieszczonych w jednej głównej pętli.

Python test 2.py - C:/Users/admin/AppData/Local/Prog

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc=Minecraft.create()
import time
time.sleep(5)
```

**1** Zaczynamy od podstawowych konstrukcji.

**2** Następnym etapem jest pobranie pozycji naszej postaci oraz przypisanie zmiennych do koordynatów.

**3** Kolejnym krokiem jest przygotowanie pętli głównej oraz pętli składowych.

**Jeżeli popełnisz błąd w którymś ze skryptów i będziesz potrzebować pomocy, zawsze możesz zajrzeć do zawartości płyty dołączonej do książki. Znajdziesz tam gotowe projekty przygotowane przeze mnie i moich kolegów – autorów książki.**

Python test 2.py - C:/Users/admin/AppData/Local/Prog

File Edit Format Run Options Window Help

```
from mcpi.minecraft import Minecraft
mc=Minecraft.create()
import time
time.sleep(5)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z
```

Konstrukcja będzie wyglądała tak:

**Pętla główna**
- **pętla przód**
- **pętla prawo**
- **pętla tył**
- **pętla lewo**

Pojedynczo pętle będą tworzyć tylko część spirali. Zadaniem głównej pętli będzie powtarzanie działań pętli znajdujących się w jej wnętrzu. W ten sposób będziemy w stanie stworzyć spiralne schody.

```
Python test 2.py - C:/Users/admin/AppData/Local/Programs/
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc=Minecraft.create()
import time
time.sleep(5)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z

for i in range(10):
```

Zaczynamy od konstrukcji pętli głównej. Następnie konstruujemy pętle składowe:

**Przód**

```
for przod in range(1,3):
    mc.player.setPos(x+przod,y+przod,z)
    mc.setBlock(x+przod,y+przod-1,z,1)
    time.sleep(0.1)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z
```

**Prawo**

```
for prawo in range(1,3):
    mc.player.setPos(x,y+prawo,z+prawo)
    mc.setBlock(x,y+prawo-1,z+prawo,1)
    time.sleep(0.1)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z
```

**Tył**

```
for tyl in range(1,3):
    mc.player.setPos(x-tyl,y+tyl,z)
    mc.setBlock(x-tyl,y+tyl-1,z,1)
    time.sleep(0.1)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z
```

**Lewo**

```
for lewo in range(1,3):
    mc.player.setPos(x,y+lewo,z-lewo)
    mc.setBlock(x,y+lewo-1,z-lewo,1)
    time.sleep(0.1)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z
```

Ostatecznie cały skrypt będzie wyglądał następująco:

```
Python test 2.py - C:/Users/admin/AppData/Local/Programs/Pytho
File  Edit  Format  Run  Options  Window  Help
from mcpi.minecraft import Minecraft
mc=Minecraft.create()
import time
time.sleep(5)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z

for i in range(10):
    for przod in range(1,3):
        mc.player.setPos(x+przod,y+przod,z)
        mc.setBlock(x+przod,y+przod-1,z,1)
        time.sleep(0.1)
    poz=mc.player.getPos()
    x=poz.x
    y=poz.y
    z=poz.z

    for prawo in range(1,3):
        mc.player.setPos(x,y+prawo,z+prawo)
        mc.setBlock(x,y+prawo-1,z+prawo,1)
        time.sleep(0.1)
    poz=mc.player.getPos()
    x=poz.x
    y=poz.y
    z=poz.z

    for tyl in range(1,3):
        mc.player.setPos(x-tyl,y+tyl,z)
        mc.setBlock(x-tyl,y+tyl-1,z,1)
        time.sleep(0.1)
    poz=mc.player.getPos()
    x=poz.x
    y=poz.y
    z=poz.z
```

Skrypt można uruchamiać kilkakrotnie, by stworzyć jeszcze wyższe schody, jednak pamiętaj, że granica świata Minecrafta pod względem wysokości kończy się na 256 bloku. Jeżeli spróbujemy przedostać się wyżej, nasza postać zacznie otrzymywać obrażenia, a bloki budowane ponad granicą świata nie będą widoczne.

**Przerwy przed niektórymi funkcjami są bardzo istotne. Nazywamy je tabulacją. Pozwalają one na uporządkowanie programu. Dzięki nim pętle prawo, lewo, góra, dół „wiedzą", że podlegają konkretnej pętli. Bez tabulacji program nie wykonywałby się prawidłowo.**

# Jak znaleźć błąd w skrypcie?

Czasami się zdarza, że podczas pracy ze skryptem wprowadzimy błąd. Może to być błąd z pozoru drobny, taki jak:

- literówka
- brak czy nadmiar liter lub cyfr
- powtórzenie tekstu

Zdarzają się również błędy dotyczące struktury, jak niewłaściwie użyta tabulacja. Na szczęście wiele środowisk programowania oferuje programistom pomoc, między innymi pokazując im błędy, a nawet podpowiadając, na czym błędy polegają.
Spróbujemy teraz w skrypcie spiralnych schodów dokonać kilku zmian, które będą miały na celu wprowadzenie błędów do naszego skryptu.

Wprowadzone błędy to:

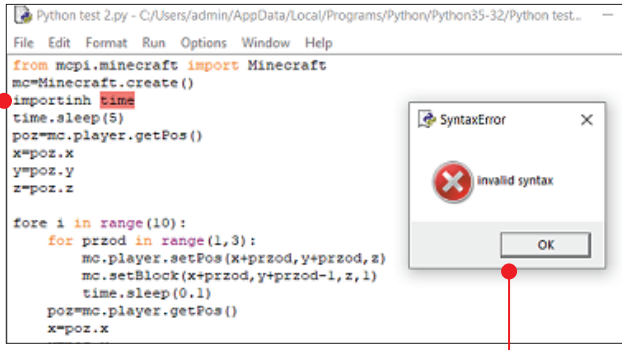
- zmiana słowa import na importinh **A**
- zmiana funkcji pętli for na fore **B**
- zmiana tabulacji w 21 wierszu **C**
- zmiana tabulacji w 31 wierszu **D**

Teraz uruchamiamy taki kod i zobaczmy, co się wydarzy.

*Python test 2.py - C:/Users/admin/AppData/Local/Programs/Python/Py

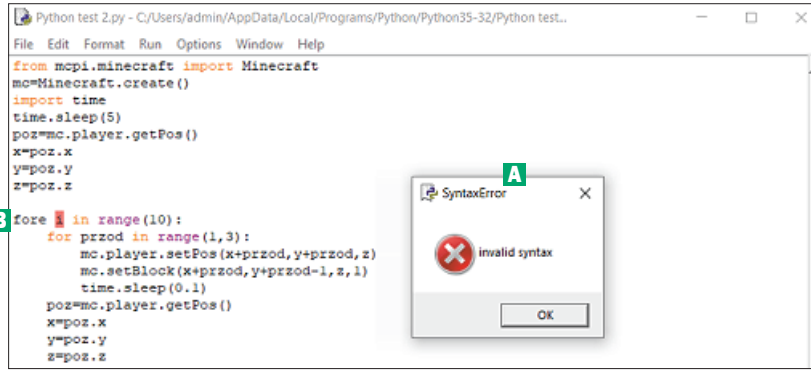
File Edit Format Run Options Window Help

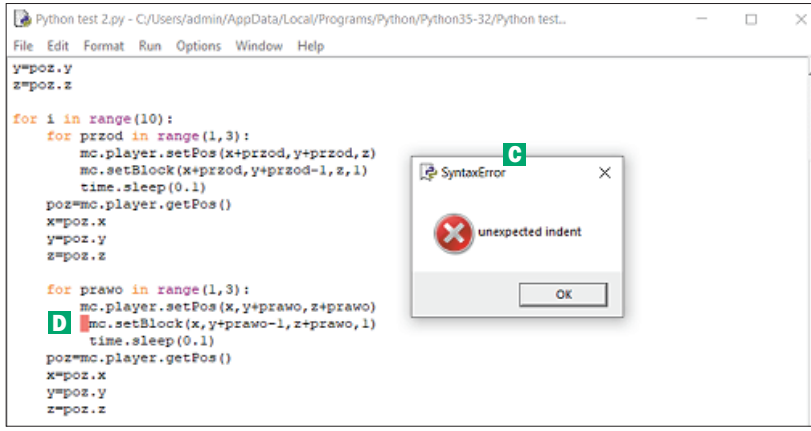
```
from mcpi.minecraft import Minecraft
mc=Minecraft.create()
A importinh time
time.sleep(5)
poz=mc.player.getPos()
x=poz.x
y=poz.y
z=poz.z

B fore i in range(10):
    for przod in range(1,3):
        mc.player.setPos(x+przod,y+przod,z)
        mc.setBlock(x+przod,y+przod-1,z,1)
        time.sleep(0.1)
    poz=mc.player.getPos()
    x=poz.x
    y=poz.y
    z=poz.z

    for prawo in range(1,3):
C       mc.player.setPos(x,y+prawo,z+prawo)
        mc.setBlock(x,y+prawo-1,z+prawo,1)
        time.sleep(0.1)
    poz=mc.player.getPos()
    x=poz.x
    y=poz.y
    z=poz.z

    for tyl in range(1,3):
        mc.player.setPos(x-tyl,y+tyl,z)
D       mc.setBlock(x-tyl,y+tyl-1,z,1)
        time.sleep(0.1)
    poz=mc.player.getPos()
    x=poz.x
    y=poz.y
    z=poz.z

    for lewo in range(1,3):
        mc.player.setPos(x,y+lewo,z-lewo)
        mc.setBlock(x,y+lewo-1,z-lewo,1)
```




Syntax error jest to jeden z najczęstszych błędów. Oznacza on błędną składnię kodu – program nie rozumie części kodu. Rzadko się zdarza, by udało się uruchomić kod, który zawiera tego typu błędy.

Wyświetliła się informacja o błędzie, został też wyróżniony wyraz po pierwszej wprowadzonej przez nas zmianie ●. Komunikat brzmi: **Invalid syntax** ●.
Po naprawieniu zmiany próbujemy wywołać kod ponownie.



```
Python 3.5.1 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec  6 2015, 01:38:48) [MSC v.1900 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:/Users/admin/AppData/Local/Programs/Python/Python35-32/Python test 2
.py
Traceback (most recent call last):
  File "C:/Users/admin/AppData/Local/Programs/Python/Python35-32/Python test 2.p
y", line 5, in <module>
    poz=mc.player.getPos()
  File "C:\Users\admin\AppData\Local\Programs\Python\Python35-32\lib\site-packag
es\mcpi\minecraft.py", line 66, in getPos
    return CmdPositioner.getPos(self, [])
  File "C:\Users\admin\AppData\Local\Programs\Python\Python35-32\lib\site-packag
es\mcpi\minecraft.py", line 32, in getPos
    s = self.conn.sendReceive(self.pkg + b".getPos", id)
  File "C:\Users\admin\AppData\Local\Programs\Python\Python35-32\lib\site-packag
es\mcpi\connection.py", line 63, in sendReceive
    return self.receive()
  File "C:\Users\admin\AppData\Local\Programs\Python\Python35-32\lib\site-packag
es\mcpi\connection.py", line 57, in receive
    raise RequestError("%s failed"%self.lastSent.strip())
mcpi.connection.RequestError: b'player.getPos()' failed
>>>
```

Komunikat **Syntax error** A pojawił się znowu, tym razem przy drugiej zmianie wprowadzonej do kodu B. Po jej cofnięciu wywołujemy kod po raz kolejny.

Tym razem pojawia się inny komunikat – **Unex pected indent** C, czyli nieprawidłowe wcięcie. Program zauważył nieprawidłową tabulację i zaznaczył ją dla nas D. Podobnie będzie w przypadku ostatniej wprowadzonej zmiany.

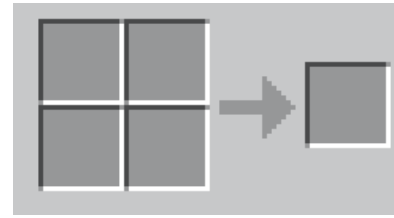
Dopiero po wprowadzeniu poprawek skrypt wykona się prawidłowo.

W przypadku gdy skrypt będzie prawidłowy, ale podczas jego wykonywania nastąpi jakiś błąd, jak na przykład brak włączonego serwera, nieprawidłowo zaimportowane biblioteki itp., konsola poinformuje nas o tym, że nie udało się właściwie wykonać skryptu. Wyświetli nam jednak w tym celu inne okno konsoli.

**Pamiętaj – czerwony kolor nie bez powodu jest kolorem oczu pająka. Prawie zawsze zwiastuje niebezpieczeństwo.**
**O, moja lama wreszcie się odnalazła i mogę wyruszyć w dalszą podróż. Bardzo ci dziękuję za towarzystwo i życzę powodzenia w dalszych przygodach z programowaniem.**
**Może się jeszcze spotkamy.**

# Warto wiedzieć

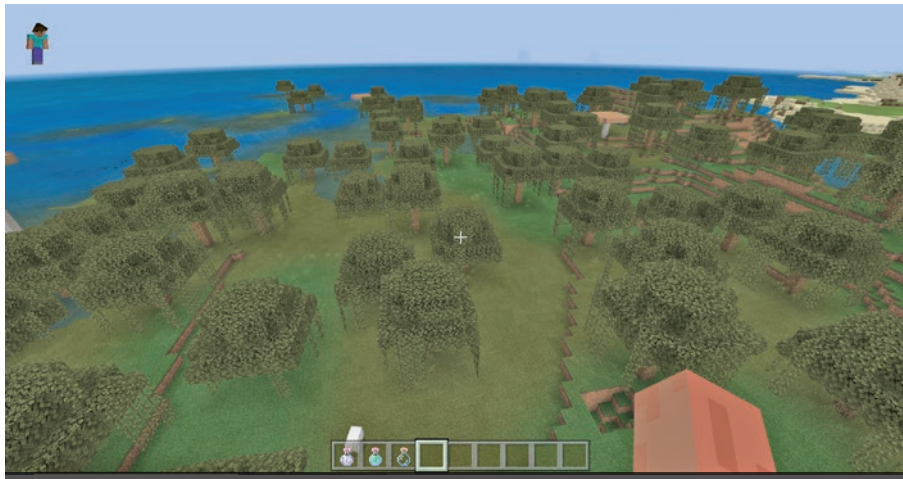
■ **Biom** – wydzielony w grze obszar z charakterystycznymi cechami odróżniającymi go od innych terenów: krajobrazem, florą, klimatem, wysokością, występującymi w nim stworzeniami itp. Obecnie istnieje około 50 różnych typów biomów (wliczając w to również biomy przejściowe).



■ **Blok** – podstawowy budulec świata Minecrafta, który jest jego cechą wyróżniającą. Ma rozmiar 16 x 16 pikseli. Większość bloków jest statyczna, choć występują też bloki ruchome. Najczęściej bloki nie podlegają prawom grawitacji (z pewnymi wyjątkami). Bloki mogą być wydobywane i przetwarzane przez gracza na wiele sposobów, co stanowi istotę gry.

■ **Crafting** – wytwarzanie przedmiotów za pomocą różnych materiałów. Aby dokonać craftingu, należy mieć wszystkie elementy składowe przedmiotu oraz znać jego recepturę. Bardziej skomplikowane przedmioty wymagają posiadania Crafting Table, czyli Stołu rzemieślniczego.

■ **Crafting Table** – jedno z najczęściej wykorzystywanych narzędzi w grze. Dzięki niemu jesteśmy w stanie wytworzyć wiele skomplikowanych przedmiotów i receptur. Jest to jeden z pierwszych bloków tworzonych przez gracza w grze. Do jego stworzenia wystarczy jedynie drewno.



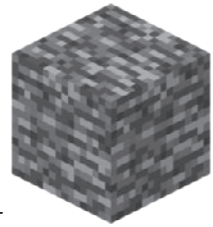
Przykład biomu bagien

Nether, Otchłań, piekielny świat; są tu biomy takie jak: Dolina Dusz, Netherowe Pustki czy Wypaczony Las

■ **End** – jeden z trzech podstawowych wymiarów Minecrafta (obok świata podstawowego, czyli Powierzchni lub po prostu Świata, oraz Netheru). Znajdziemy tu głównego bossa gry – Smoka Endu. Jest to królestwo Endermanów.

End – królestwo Endermanów

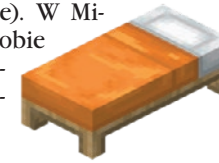
■ **Nether** – inaczej nazywany Otchłanią. Jest to jeden z trzech podstawowych wymiarów Mine crafta, w którym znajdziemy olbrzymie złoża netherracku (to blok budujący Nether). Są tu różne stworzenia niewystępujące w innych wymiarach, jak Ghasty czy Witherowe Szkielety.



■ **Skała macierzysta** – najbardziej wytrzymały blok występujący w świecie Minecrafta. Stanowi podłoże świata podstawowego oraz górną i dolną granicę Netheru. Według twórców gry nie da się go zniszczyć w trybie innym niż Kreatywny.

■ **Skin** – tekstura wyglądu gracza i wyglądu innych postaci (mobów).

■ **Spawn point** – miejsce, w którym pojawia się gracz po rozpoczęciu gry lub po śmierci (odrodzenie). W Minecrafcie można sobie ustawić punkt odrodzenia poprzez przespanie się w łóżku.



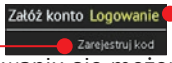
■ **Steve** – główny bohater gry. Jeden z dwóch wariantów (drugim jest Alex – postać kobieca). Jednocześnie jest to podstawowa skórka postaci (którą można zmienić).

■ **XP** – punkty doświadczenia, pełnią funkcję pomocniczą i przydają się na przykład do zaklinania przedmiotów.

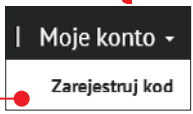
# JAK SKORZYSTAĆ Z E-WYDANIA KSIĄŻKI

W KŚ+ znajdziemy **e-wydanie tej Biblioteczki, obraz ISO dołączonej do niej płyty z narzędziami programistycznymi i plikami projektów oraz plik PDF książki do pobrania.**

1 Otwieramy stronę **ksplus.pl**. Logujemy się (używamy konta z serwisu **Komputerswiat.pl**). Jeżeli nie mamy konta, klikamy na, by się zarejestrować.




2 Po zalogowaniu się możemy zarejestrować kod nadrukowany na płycie do łączonejdo książki. Wystarczy kliknąć na link i przepisać kod.




3 Uzyskamy w ten sposób dostęp do e-wydania i do bonusowego obrazu płyty. Do serwisu KŚ+ możemy logować się z dowolnego urządzenia z dostępem do internetu.

**UWAGA! W KŚ+ ZA DARMO E-WYDANIE KSIĄŻKI ORAZ PLIK ISO PŁYTY**

# POLECAMY INNE NASZE KSIĄŻKI

## GIMP KROK PO KROKU

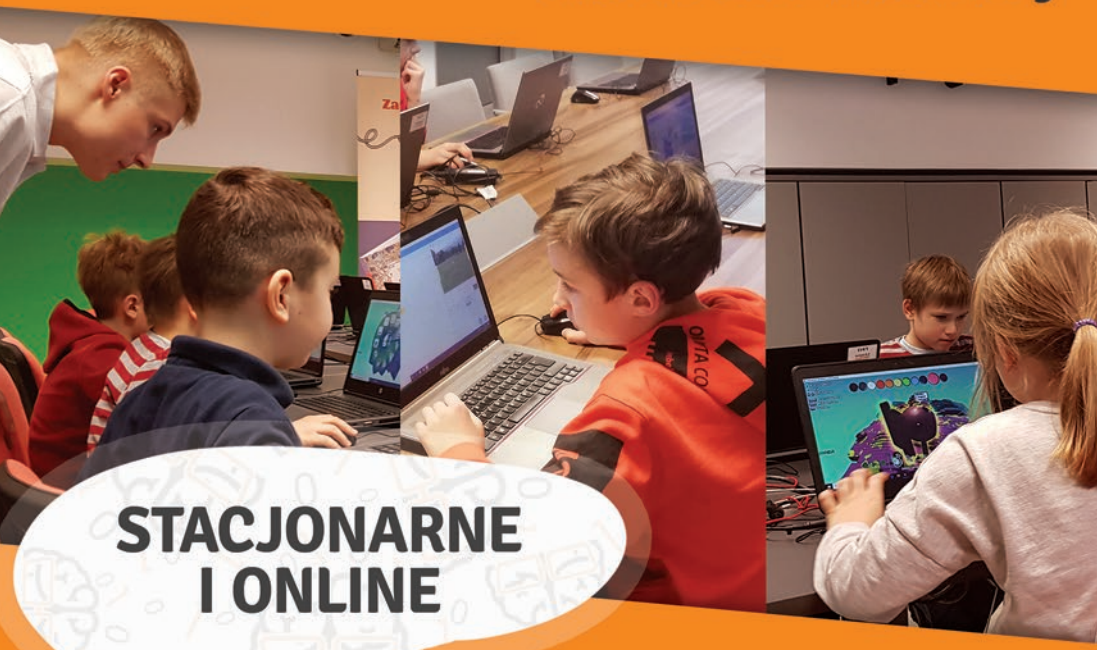
Kurs obsługi najlepszego darmowego edytora zdjęć i grafiki: jak korzystać z najważniejszych narzędzi do retuszu, fotomontaży, efektów i poprawiania fotek. Na DVD: 30 najlepszych narzędzi graficznych.

## WSZYSTKO O WINDOWS

Krok po kroku, jak pobrać i zainstalować Windows, rozwiązać problemy z systemem, dbać o prywatność i zorganizować pracę zdalną. Na DVD: 50 najlepszych programów do Windows.

**Nasze książki kupisz na literia.pl/literia-poleca/komputer-swiat**
**Książki są również dostępne w wersji elektronicznej na ksplus.pl**

# KURSY PROGRAMOWANIA
## dla dzieci i młodzieży



## STACJONARNE I ONLINE

## WYSTARTOWAŁY!
## zapisy na SEMESTR JESIEŃ/ZIMA!

- Twórz własne gry w Minecrafcie!
- Zaprogramuj swoją stronę internetową
- Zacznij tworzyć swoje aplikacje

MINECRAFT ROBLOX SCRATCH unity C# Lua

www.giganciprogramowania.edu.pl

tel: 22 11 210 63

giganci programowania

Autorzy (od lewej): Radosław Kulesza, Sebastian Langa, Dawid Leśniakiewicz, Adrian Czechowski, Piotr Pełka – współtwórcy szkoły Giganci Programowania, oferującej kursy programowania dla dzieci i młodzieży

giganci programowania

# CZWARTY WYMIAR MINECRAFTA

Minecraft to ogromnie popularna gra. Niesamowite jest to, że może stanowić nie tylko rozrywkę, ale też platformę do nauki programowania – bardzo atrakcyjną, bo efekty naszych programistycznych prób możemy obserwować bezpośrednio w grze. I o tym właśnie jest ta książka – o programowaniu jako czwartym wymiarze Minecrafta, obok znanych wszystkim trzech podstawowych wymiarów, jakimi są Świat, Nether i End. Umiejętność programowania daje władzę nad całym światem gry i pozwala go dowolnie zmieniać.

Dzięki tej książce poznamy najważniejsze sposoby na programowanie świata Minecrafta według własnych pomysłów. Każdy rozdział zapoznaje czytelnika z takimi narzędziami, jak MakeCode, MCreator i język Python oraz z ważnymi programistycznymi zagadnieniami. Programowania uczymy się, tworząc własne projekty przedstawione krok po kroku we wskazówkach.

Książka została napisana przez specjalistów od nauczania programowania – wykładowców szkoły Giganci Programowania, którzy na co dzień prowadzą kursy tworzenia gier, stron internetowych i aplikacji.

Na płycie dołączonej do książki znajdziemy najlepsze narzędzia programistyczne oraz pliki opisanych projektów (ISO płyty można również pobrać z ksplus.pl).

**CENA 16,90 ZŁ**
W TYM 5% VAT

Płyta DVD jest dodatkiem do książki

ISBN 978-83-8091-952-5 INDEKS 321 958 05

Nr 5/2020 (109)

KOMPUTER ŚWIAT BIBLIOTECZKA